

信息化的科学梦

从计算机科学到信息化科学

屈延文 编著

電子工業出版社

Publishing House of Electronics Industry

北京 • BEIJING

内 容 简 介

当代信息化事业的发展，既需要以“产业为中心”的计算机科学，也需要“用户为中心”的信息化科学，信息化科学与计算机科学为整个信息化构成两个中心的新型的“太极”理论体系。本书对计算机科学的发展过程进行了回顾，并介绍了信息化服务性理论、信息化安全性理论、信息化监管理论、信息化认证理论、人类信息化活动行为学、信息化总体学与体系结构、可信网络世界体系结构框架（TCAF）、信息化技术法规、信息化评估理论、信息化测评认证理论和信息化科学使命等诸多方面的内容，以建立独立的信息化科学体系。本书内容权威，讲解深入，是信息学科硕士、博士的必读书籍，也可以供所有从事信息化发展相关工作的人员参考。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有，侵权必究。

图书在版编目（CIP）数据

信息化的科学梦：从计算机科学到信息化科学 / 屈延文编著. —北京：电子工业出版社，2006.6
ISBN 7-121-01271-5

I.信... II.屈... III. 信息技术—研究 IV.G 202
中国版本图书馆 CIP 数据核字（2006）第 065051 号

责任编辑：牛晓丽

印 刷：

装 订：

出版发行：电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

开 本：140×203 1/32 印张：11.875 字数：304 千字

印 次：2006 年 6 月第 1 次印刷

印 数：1630 册 定 价：55.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlt@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

序

本书的作者屈延文教授将他的论著《信息化的科学梦——从计算机科学到信息化科学》的清样送给我，请我写个序言，我原以为这是一本讲述计算机科学发展过程的书。初步翻阅全书后，才发现它是一本面向未来的计算机与信息化科学的理论性书籍。作者在几十年从事计算机科学、计算机系统研制、程序设计语言与操作系统和信息化工作的长期研究与实践基础上，提出“信息化科学”这一命题，是颇有创意和勇气的。当然，一种新的学说的创立是不容易的，往往不是单独可毕其功。作者在书名中使用“梦”这个字，表达他对“信息化科学”的发生与发展的展望与憧憬，希望关怀信息化事业的广大学者和工作者能给予重视并积极参与研究。

本书系统地阐述了计算机科学发展与信息化科学的新思想和新观点，并提出了我国信息化要从当前初级阶段走向高级阶段必须实现科学发展，必须通过建立信息化科学体系来指导信息化事业发展的主张。

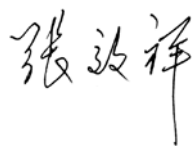
信息化已成为我国覆盖现代化建设全局的战略举措，信息化也是当今世界经济社会发展的大趋势。信息化应是人类共同的事业，内涵丰富，耗资巨大，过程很长。对信息化大量的实践和问题，理应从感性认识提高到理性认识，发现并梳理信息化本身所体现的一系列规律和理论问题，以指导人们更自觉和科学地开展信息化事业，这可能是“信息化科学”的首要使命。

信息化的全球性发展，使人类逐步迈入信息社会时代，使人类生活、生产活动从自然空间扩展到网络空间，在这样的新环境里引发的人际关系、伦理形态、生活与生产方式、道德与行为准

IV ▶▶▶ ■ 信息化的科学梦——从计算机科学到信息化科学

则、文化特征、法制体系等许许多多方面的变化，将成为人类必然要面对的新问题。这些问题似乎也应归入“信息化科学”的研究范畴，因此“信息化科学”可能是一种横跨自然科学和社会科学的综合性科学。

衷心祝愿“信息化科学”顺利成长和深入发展！

A handwritten signature in black ink, reading '张波祥' (Zhang Bixiang). The characters are written in a cursive, flowing style.

2006年4月24日

德国数学家莱布尼兹学习了易经，提出了二进制，但是他仅仅模仿了中国哲学思想最表面的东西。中国信息化的理论学者不应该把可信息化的社会科学与哲学的形式化留给下一个“莱布尼兹”，而应该争当到达成功彼岸的先行者。本世纪的信息化和 IT 技术大师是信息化科学的大师。

前 言

为什么要提出信息化科学？在当代信息化发展中，电子政务、电子商务、人民生活信息化和国家基础设施信息化等在传统的信息化推进模式（红头文件+资金投入）中变得十分困难，新世纪的信息化面对的不是具体的业务信息系统，而是研究信息化的运营模式、系统平台和技术体系结构，因为信息化越来越强调其综合效益。信息化研究甚至到了没有上层的信息化体系结构研究，就难于开展的程度。中国有强劲的信息化需求，却存在着实现信息化的困难。

传统的计算机科学面对现代信息化的发展要求，感到力不从心，失去了前期的指导和基础作用，不能适应信息化的发展。信息产业与计算机科学在早期提出的计算理论、计算机工程、软件工程、信息系统的理论和实践，曾经为信息化发展发挥过巨大作用。在大范围、大规模、超海量数据与对象以及高智能应用与管理的要求面前，在信息化越来越要求发挥综合效益的今天，虽然信息产业与计算机科学为信息化体系结构提供了标准化方法，但没有提供信息化的理论方法。大量的信息化问题没有基础理论研究支持，而具有的仅仅是“解决方案”的一个一个案例。当代信息化的特点是：有大量的信息化问题与现象，有大量信息化问题的解决方案，而没有深入的、更高层的信息化理论研究，当然也没有信息化学科和信息化科学。因此，我们说当代的信息化处在初级和混乱的阶段，只有通过建立信息化科学，以信息化科学指导信息化的发展，才能进入高级阶段。当前信息化面临的主要问题是：信息化如何科学发展？如何发挥信息化综合效益？如何加强信息化总体学与体系结构的指导作用？如何强化信息化公众服

务、互操作性、安全、监管与认证？如何强化信息化技术法规、标准、评估和测评？整个 IT 行业酝酿着重大的变革，随着信息化发展越来越脱离传统通信、计算机、软件和网络等学科研究范畴，信息化逐步产生了属于自己的独立学科：信息化科学。

近两年来，在全国许多大学讲授《软件行为学》和介绍新时代信息化与安全技术发展的展望期间，笔者了解了一些大学的研究生和博士生的学习与研究情况：许多大学的老师与学生都被捆绑在一些旧的知识上，虽然也研究一些新问题，但多数不是当前信息化需要解决的理论、体系、方法和技术方面的紧迫问题，而是一些从国外引入的不成熟的概念。这使笔者深深地感到，需要呼唤信息化科学的创立，来弥补计算机科学的不足与改变计算机科学研究的停滞状态，迅猛追赶信息化发展的脚步。在全世界的 IT 产业与信息化处于低谷的时期，我国信息化发展的需求十分强劲，我国有比任何国家更多、更复杂的信息化问题需要去研究。有志的青年信息化科学工作者应当勇挑重担，开辟信息化科学研究的新天地。

在这样的大形势下，推动信息化科学的研究与发展是带动信息化发展进入高级阶段的根本出路。在过去几十年内，计算机科学实际上是以乙方为中心的，即采用以信息产业为中心的信息化发展模式。这种模式以产业提供的产品与技术为中心来扩展应用领域。在这种认识体系中，信息化概念主要应理解为计算机应用，甲方是 IT 产业的用户。一些 IT 跨国公司提出了“服务科学”和“需求工程”，虽然认识到了当前信息化发展处于低谷的事实，但是开出的“药方”却仍然不能从根本上解决问题。信息化的发展从低级阶段到高级阶段越来越依赖于甲方，现在，应当建立以甲方为中心的发展模式。甲方如果仅仅作作为 IT 技术与产品的用户，而不发展信息化科学，将信息化应用发展到更高阶段，那么信

息化发展将会遇到根本性的困难。当代信息化事业的发展，既需要以产业为中心的计算机科学，也需要用户为中心的信息化科学，信息化科学与计算机科学为整个信息化构成了两个中心的新型的“太极”理论体系。2500年前诸子百家的敢想、敢说、敢创立新理论和新学说的智慧一定会指引中国信息化走向更光辉的未来！新时代的理论创新首先需要无比的勇气和气魄，其次需要从问题与实际出发进行理论研究的智慧和刻苦细致的工作。

信息化和信息产业的发展一般是通过开拓信息化应用与市场需求来促进的，而需求是由认识决定的，因此，提高认识水平，扩展自己的知识体系，建立新的理论及学说，就是牵动需求和促进发展。本书的目的就是通过建立新认识、新知识、新理论和新学说来促进信息化的发展。建立现代理论工作同样需要总体工作，需要对所研究的对象进行全局和结构性的构思，研究各种问题的关系，确立理论研究的框架结构。这种理论研究的总体工作，对于信息化科学尤其重要。

首先，本书献给那些希望了解各种信息化应用模式的数学家，使他们以此为启迪来研究更深层次的形式化和数学理论问题。其次，献给社会学家、经济学家、金融学家、管理学家和系统工程师，使他们把各自的方法学结合起来，把各自的关注点结合起来，建立一个统一的认识体系平台。在这个认识体系平台上，社会学家、经济学家、金融学家、管理学家、系统工程师、计算机科学家和数学家各自实现新的方法和进行实践时，会得到其他学科的支持，从而达到一个新的高度，为实现我们共同的梦想而奋斗！

笔者与通信专家孙玉院士、信息安全与密码学专家南相浩教授以及中国信息化与安全业界同仁共同筹建了信息化与安全的总体咨询机构：QNS工作室。QNS工作室以构建中国信息化发展的科学梦为己任，以建立独立的信息化科学体系为目标，联合众多

的业界人士，以新认识、新知识、新理论、新学说从事空前规模的信息化科学推动工作。

对于大多数热衷于信息化的读者来说，本书是一本长期有用的书籍，今年看有收获，明年再看还会有新收获。本书共分为 33 章，提出了 170 个信息化科学研究问题，其中的大多数问题都可以作为博士论文的选题。笔者编写本书不是为了宣扬什么，而是为了与信息化科学研究的志愿者进行交流。本书仅是在中国信息安全产业的博士论坛上的一篇开场报告，令人钟情的结果还在未来。

感谢深受敬重的张效祥院士字斟句酌地为本书写序，感谢孙玉、南相浩、林鹏，感谢吴世忠、姜广智、王贵驷、张焕国、吕述望等多年来的支持，感谢贺卫东、严立、刘毅、扬勇刚、申屠建中、曹斌等以及信息安全产业界的朋友们在长期合作中所给予的帮助，特别是贺卫东先生对本书出版与发行所做的工作。

本书也是笔者给电子计算机诞生 60 周年和中国电子计算机诞生 50 周年的献礼。

屈延文
2006 年 4 月

概 述

本书的目标是为创立信息化科学的新型学科摇旗呐喊。计算机科学对当代信息化的研究、应用与发展的指导能力不够，其理论研究必须确立为信息化发展服务的宗旨。信息化发展产生的许多现象使计算机科学家感到困惑。许多传统的计算机科学家对信息化发展出现的问题没有基本的认识，也难于理解。他们找不到信息化应当建立的模型。笔者希望计算机科学发挥在计算机诞生时代和促进计算机应用中所表现出的前瞻性优势，推动信息化科学的建立，并在其中发挥指导作用。

对于信息化而言，传统计算机科学的经典理论中有许多已经成为计算机科学史了，学习计算机科学的目的是了解当时的计算机科学家是如何在计算机诞生时期解决计算问题的，了解他们是如何在计算机应用中解决问题的。仅仅依靠图灵机理论、自动机理论、确定计算和不确定计算理论、计算复杂性理论，不能解决当代信息化面临的计算问题。例如，传统的系统概念不能再理解为自动机理论描述的字符集、状态集、状态转移、输入/输出、初始状态和终止状态的自动控制概念；现代的系统概念不仅涉及到信息基础设施和网络，规模巨大，而且人类本身已经进入到系统之中了。又例如，程序设计语言、形式语言理论、程序正确性理论、形式语义理论、软件工程理论和人际交互理论对软件开发与生产发挥过且仍然正在发挥着重要的作用。再例如体系结构理论，在早期有计算机的体系结构，后来发展到软件的体系结构，甚至网络或系统的体系结构，但是目前发展起来的是信息化体系结构，不仅仅涉及到系统体系结构和技术体系结构，同时还涉及

到运营的体系结构（人类使用信息系统行为的体系结构）。也就是说，传统的计算机科学面对现代信息化的发展要求，感到力不从心，失去了前期的指导和基础作用，不能适应信息化的发展。

本书首先回顾了计算机科学的发展历史，接着介绍了现代数学与计算机科学、计算机可读数学、信息化与形式化、计算理论、电子计算机、程序设计语言、计算机操作系统、计算机应用、网络系统与网络世界、软件工程与面向对象的程序设计、并发程序设计、面向代理（主体）的程序设计、可视化与多媒体、光学计算与光学计算机、计算机理解科学（形式语义与形式说明）、系统集成、统一建模语言（UML）与可扩展标记语言（XML）、代理化、互操作性平台理论、网络世界行为学、信息化服务性理论、信息化安全性理论、信息化监管理论、信息化认证理论、网络行为对抗理论、群体计算理论、人类信息化活动行为学、信息化总体学与体系结构、可信网络世界体系结构框架（TCAF）、信息化技术法规、信息化标准化理论、信息化评估理论、信息化测评认证理论、可信息化科学（社会科学、军事科学、自然科学、物理科学、天文学、生物科学、地理科学、能源科学、环境科学等科学领域）形式化、哲学形式化和信息化科学使命等诸多方面的内容。

那么，什么是信息化呢？

我们可以给信息化下一个定义：信息化是人类社会的一切主体运用信息技术和信息产业提供的产品与服务，经过面向发展需求的规划、综合、标准化、设计、系统工程和项目建设，实现政治、经济、生产、社会、生活、文化、军事等一切领域的运营、工作、管理、组织等模式的变革进程。简单来说，信息化是人类使用 IT 技术与产品的社会活动。信息化技术是运用 IT 技术与产品的再工程和再设计的技术。

通常意义上的通信专家、计算机专家、网络专家、微电子专家、软件专家等并不能自然地成为信息化专家，因为通信领域通常关注的是通信效率和通信质量，计算机领域通常关注的是处理能力和存储能力，计算机网络领域通常关注的是信息共享和信息交换，信息基础设施领域通常关注的是高速宽带和资源利用。信息化的问题越来越多，而且也越来越脱离传统通信、计算机、软件、网络等学科的研究范畴。把这些信息化中产生的新问题综合在一起，采用区别于传统信息技术科学的方法进行研究，就形成了一个新学科，我们把这个新学科称为“信息化科学”。这个新学科关注的是互操作性、安全性、服务性、用户标准、技术法规、测评、认证、监管、配置、管理和服务等新问题。信息化科学逐步产生了自己独立的理论学科体系。

信息化科学是人类信息化行为科学、网络行为科学与计算机科学组合与互动产生的新型学科。

信息化科学与传统科学的区别是什么？当今世界至少存在着三种科学体系：

- ◆ 经典科学（物理、化学、生物学和电子学等） 经典科学理论正确与否是通过实验来验证的，理论的验证者通常在被研究对象之外（除非极其微小的研究对象，否则，实验仪器对对象会有一定的影响），验证活动是人类认识世界的活动，被研究的对象可以被认为是纯客观性的。
- ◆ 数学科学 最基础的数学是唯物产生的，有直接的直观真理意义，但是，现代数学基本上已演变为一堆抽象的结构，具有唯心、主观和人造的特性。虽然每一个数学结构都建立在自己的公理体系上，但这些公理是任意的、人造的和纯粹的。这些数学的正确性是通过证明的方法来确认和验

证的，确认与具有直觉真理意义的数学是一致的、无悖论的。显然，数学理论的验证者在形式系统之外，这种验证活动是人类的推理活动。

- ◆ 应用科学 尤其是信息化应用科学，其理论的正确性是通过应用来验证的，人类通常处于信息化应用系统之内，其验证活动是人类利用信息化活动本身进行的。显然，应用科学与经典科学及数学是不相同的。当前，信息化应用科学的重点是研究人类利用信息化的行为和网络世界中虚拟主体的行为以及它们之间的关系。

信息化科学涉及经典科学（电子学、磁学、光学、半导体学、电子计算机和光学计算机等），也涉及数学科学。例如，软件领域中主要涉及的是数学，这是传统计算机科学的研究领域。当然，信息化科学也涉及应用科学，这是当代信息化科学的研究重点。在信息化科学中，需要研究人类信息化行为和网络世界中虚拟主体的行为学理论，也需要研究软件行为学、行为形式逻辑学和事物变易原理的形式逻辑学。

中国人研究行为学具有最久远的历史，虽然那时不叫行为学，而叫做易经、道德经、礼仪、儒学和兵法。由于这些科学非形式化，不能反复试验和重演，所以必须仔细体会和领悟，才能理解它们。在这方面，社会科学与自然科学在人类接受的程度上差别极大。

对于自然科学的认识，通过教育已经或在相当程度上解决了知识的继承性和发展的问题，那些早期自然科学最前沿的成果中有些现在已经成为大学甚至中学的教育内容，它们很容易被受教育者接受，因为受教育者对接受前辈的研究成果没有抵触情绪。

但是，对社会科学的认识与学习就没有这么简单了。在向学

生教授社会科学的成果时，很容易受到抵触。虽然 2500 多年前提出的社会科学方法与结论依然被人们当做神圣的东西来传授，儒学、孙子兵法等都是如此；虽然也有人利用“三字经”等强迫不懂事的孩子去背那些先人的研究成果与结论，但依然不能将人类社会科学的早期研究成果转变成学生可以理解的知识。所以，社会科学在认识论上有一个公理，这就是孔子的名言：吾十有五而志于学，三十而立，四十而不惑，五十而知天命，六十而耳顺，七十而从心所欲，不逾规。当人们体会到社会科学中的许多理论概念时，也已经到了老年。从出生到死亡，人对社会的认识就像锯齿波一样，随着人出生、成长和死亡的周期而起伏。社会科学没有形成一个形式系统供人们反复试验，来验证它的正确性（当然也是形式化正确的验证过程）。但是，社会科学教育与认识的可试验和重演，在信息化和网络化的时代变得可能了，其出路就是不断地完善信息化社会科学的形式化。当然，这种形式化必须是计算机可读的。

目 录

第 1 章 信息化科学梦的浮出	1
1.1 计算机科学发展回顾	2
1.2 现代数学与计算机科学	5
1.3 计算机可读数学	9
1.4 信息化与形式化	22
第 2 章 计算理论	30
2.1 可计算性理论	31
2.2 算法理论	32
第 3 章 电子计算机	34
3.1 单机、多机到网络虚拟计算机	35
3.2 高阶无穷计算机	36
第 4 章 程序设计语言	38
4.1 300 多种程序设计语言	39
4.2 永恒的 C 语言	41
第 5 章 计算机操作系统	42
5.1 单机与多机操作系统	43
5.2 应当开发多代理网格操作系统	45
第 6 章 计算机应用	47
6.1 可视化与代理化的管理信息系统 (Agent MIS)	48
6.2 管理、监控系统和自动化控制系统	49

6.3	可视化与代理化服务技术的融合	51
第 7 章	网络系统与网络世界	52
7.1	走向多网融合的网络世界	53
7.2	可信信息基础设施网络的基本要求	55
第 8 章	软件工程与面向对象的程序设计	57
8.1	面向对象的程序设计 (OOP)	59
8.2	面向模式的程序设计 (MOP)	60
8.3	面向类型的程序设计 (TOP)	61
8.4	面向视角的程序设计 (Aspect-OP)	78
第 9 章	并发程序设计与面向代理 (主体) 的程序设计 . .	80
9.1	传统的并发程序设计	81
9.2	类型程序设计方法支持并发程序设计	87
9.3	面向代理 (主体) 的程序设计 (Agent-OP) . .	89
9.4	类型程序设计支持面向代理的程序设计	91
第 10 章	可视化与多媒体	95
第 11 章	光学计算与光学计算机	98
第 12 章	计算的理解科学 (形式语义与形式说明) . . .	100
12.1	形式语言理论	101
12.2	形式语义学与形式说明	102
第 13 章	系统集成	106
13.1	微软公司的软件集成平台 OLE	107
13.2	DII COE 段开发技术是更高层次的系统集成技术	111



第 14 章	统一建模语言 (UML) 与可扩展标记语言 (XML)	113
14.1	统一建模语言 (UML)	114
14.2	可扩展标记语言 (XML) 与 Web 技术	116
第 15 章	代理化	118
15.1	什么是代理技术	119
15.2	代理技术的基本特性	121
15.3	代理软件工程框架	124
15.4	代理技术工厂	133
第 16 章	互操作性平台理论	136
16.1	互操作性概念	137
16.2	互操作性的实现技术	140
16.3	互操作性带来的服务模式的改变	145
16.4	互操作性引发的安全问题	145
16.5	互操作性的测评认证问题	147
第 17 章	网络世界行为学	150
17.1	什么是网络世界 (Cyber)	151
17.2	如何研究网络世界的行为学	152
17.3	网络世界行为语义描述方法的总要求	158
17.4	软件行为学是研究网络世界行为的语义学	159
第 18 章	信息化服务理论	172
18.1	网络世界的服务行为模式	173
18.2	代理化的网络服务	176
第 19 章	信息化安全理论	178
19.1	从关注网络安全到更加关注网络运营的安全	180

19.2	TCP 从不可信开启的可信网络世界安全新模式	185
19.3	建立网络世界的独立安全理论体系	188
第 20 章	信息化监管理论	195
20.1	从监管业务进网谈起	196
20.2	要建立网络世界风险监管的多视角认识体系	197
20.3	用信息资产风险监管来构建监管体系核心理念	200
20.4	网络世界风险监管理论体系	201
20.5	行为监管的形式化描述	207
第 21 章	信息化认证理论	210
21.1	公众服务认证需要组合密钥 (CPK)	211
21.2	活性认证理念带来的变革	212
21.3	行为认证形式化描述	218
第 22 章	网络行为对抗理论	222
22.1	建立网络对抗的威慑力量	223
22.2	网络行为对抗理论	225
22.3	行为对抗形式化描述	228
22.4	网络虚拟军队主体行为结构描述	230
22.5	行为对抗的态势评估	237
第 23 章	群体计算理论	239
23.1	群体计算将改变计算理论的面貌	241
23.2	群体计算的行为分析器	244
第 24 章	人类信息化活动的行为学	250
24.1	人类信息化行为	251
24.2	可信息化条件讨论	255



第 25 章	信息化总体学与体系结构	257
25.1	体系结构研究实践及存在的问题	258
25.2	体系结构理论研究	266
第 26 章	可信网络世界体系结构框架 (TCAF)	276
26.1	可信网络世界的可信概念	277
26.2	可信网络世界的平台概念	278
26.3	可信网络世界体系结构框架 (TCAF)	279
第 27 章	信息化技术法规	286
第 28 章	信息化标准化理论	289
28.1	标准需要理论与体系结构的研究	290
28.2	用户标准体系	295
28.3	现代标准化是一种网络服务体系	296
第 29 章	信息化评估理论	297
29.1	信息化工作的价值评估是核心	298
29.2	评估有效性的再评估	298
第 30 章	信息化测评认证理论	302
30.1	信息技术的安全测评认证	304
30.2	测评认证的网络化服务	320
第 31 章	可信息化科学的形式化	323
31.1	计算是现代科学研究的第三支柱	324
31.2	可信息化科学的形式化关系到人类整个知识体系的世代传承伟业	325

第 32 章	哲学形式化·	330
32.1	网络世界的哲学体系·	331
32.2	中国哲学范畴最有希望发展为网络世界的哲学体系	332
第 33 章	信息化科学使命·	334
附录 A	研究问题清单·	339
附录 B	资源·	352
参考文献	·	354

第 1 章

信息化科学梦的浮出

1.1 计算机科学发展回顾

计算机科学研究在当时对计算机的诞生和计算机的应用发挥了极其重要的指导作用。从产业发展角度看，主要在计算理论、计算机工程（如何做计算机）、软件工程（如何编软件）和信息系统与应用（如何建立信息系统）四个方面做出了巨大贡献；从科学角度看，主要表现在计算理论、计算的理解技术、计算机应用科学等几个方面；从智能化发展角度看，计算机科学走过了计算理论、理解技术理论阶段，其中，可视化技术获得了圆满的结果。现在，进入了代理化的发展阶段，并逐步走向更高智能化的发展阶段。

在计算理论为主的研究阶段，主要有自动机理论、图灵机理论、递归函数论、冯·诺依曼计算机模型理论、谓词演算模型、代数模型理论和算法理论等。这些理论开始诞生于 1936 年，其中包括图灵机模型、Godel 等人提出的递归函数论和冯·诺依曼计算机模型理论。1941 年，Church 提出了 Lambda 演算；1958 年，H. K. Curry 提出了组合逻辑理论等。

在计算的理解技术研究阶段，主要有程序设计语言、形式语言理论、程序正确性理论、形式语义理论、软件的数学理论、软件工程理论、体系结构理论、计算机图形学、计算机图像学和人际交互理论与技术研究，其中包括作为计算理解技术的可视化与多媒体技术。在这个发展过程中，美国计算机科学家 J. Buckus 提出了第一个高级程序设计语言：FORTRAN 程序设计语言。1960 年，欧洲科学家提出了另一个高级程序设计语言：ALGOL 60；1964 年，Landin 和 G. Plotkin 第一次提出为程序设计语言编写形式语义概念的操作语义。1966 年，J. Buckus 提出了著名的巴克斯范式；

同时, 计算机的形式语言理论也得到了深刻的发展, 计算机语言的编译程序设计有了坚实的理论支持。也是在 20 世纪 60 年代后期, E. W. Floyd 提出了程序的部分正确和完全正确概念, 为程序正确性研究奠定了基础。1967 年, E. W. Dijkstra 提出了 Simula-67 的 Class 技术, 开辟了 OOP 的先河。1968 年, D. E. Knuth 提出了属性文法, 对编译程序的语义生成提供了科学方法。1969 年, 英国科学家 C. A. R. Hoare 提出了程序部分正确公理系统, 使程序正确性验证有了基础。1972 年, Z. Manna 对递归函数不动点理论进行了详细的论述; 也是在这个时候, 美国科学家 McCarthy 提出了 LISP 语言, 开辟了函数程序设计的先河。1972 年, D. Scott 和 C. Strachey 提出了指称语义方法, 使形式语义学更加完美。1975 年, E. W. Dijkstra 提出了程序最弱前置条件公理语义方法。在这个时期, UNIX 操作系统诞生了, 具有编写系统软件能力的 C 程序设计语言问世了。到了 1976 年, TCP/IP 协议的计算机网络诞生。同年, Bell, D. E 和 LaPadula, L. J 提出了计算机安全状态模型。1978 年, 美国计算机科学家 J. Buckus 在他的图灵奖演说中提出了著名的 FP (Functional Programming) 理论体系。德国数学家 P. Martin_Löf 分别在 1975 年、1979 年和 1990 年提出和完善了其类型理论 (构造数学) 体系。1980 年, 美国国防部提出了 ADA 程序设计语言; 同时, C++ 程序设计语言诞生。1983 年, 世界上最好的操作系统 VAX/VMS 问世。1993 年, 美国微软公司推出了 Windows NT 操作系统和其他 Windows 产品, 并推动了可视化技术发展。当时, 可视化技术同样冲击了传统的计算机科学, 我们称可视化为计算机发展历史上的第二次革命。可视化在网络应用发展推动下达到了一个令人满意的程度。

从上述的讨论中可以看出, 计算机科学一直对计算机及其应用的发展发挥着重要的指导作用。但是最近 10 多年来, 计算机科

学对信息化发展的指导作用减弱了。如何才能重振计算机科学对信息化的指导作用呢？

从 20 世纪 90 年代中期开始，信息化发展中遇到的问题发生了很大变化，系统集成、多网融合、互操作性、网络远程服务、互联网广泛应用、计算机病毒、网络黑客、信息安全、网络安全、基础设施安全、打击网络犯罪、打击网络恐怖主义、打击网络邪教、打击信息诈骗、网络游戏、可视化内容、信息化监管、身份认证、网络对抗战争、用户技术标准、技术法规、信息化评估、信息化测评认证、信息化体系结构和信息安全应急体系建设等纷纷问世。对于上述的信息化应用潮流，传统的计算机科学一时变得不知所措。这个时候出现的主流技术是系统集成技术、互操作性技术、代理技术、引擎技术、数字标签技术、安全技术、监管技术、认证技术、体系结构技术、平台技术、群体计算技术和网络格技术。即便是标准化概念，也从过去主要关注企业的产品与技术标准转变为更加关注互操作性、体系结构等体现甲方概念的用户标准。在大范围网络环境中，面对超海量数据与对象以及高智能应用与管理要求，虽然曾经不断提出分布式计算（DCE）、网络计算平台和网络计算机等局域网发展概念，虽然曾经出现防火墙、入侵检测和将网络划分为以内外局域网为中心的安全概念，但是在信息化和全球化发展的推动下，信息化的发展道路必须毅然跳出这些从局域网发展而来的限制，产生以代理和多代理技术为核心的新型群体计算模型。我们称这个新信息化时代是代理化时代。笔者认为代理化是计算技术的第三次革命。

在信息化发展浪潮中，计算机科学的研究对象也要逐步转移到行为和内容上。需要研究互操作性理论、网络中的服务理论、安全性理论、监管理论、认证理论、网络对抗理论和体系结构理论。在这样的信息化大背景中，提出研究软件行为学、网络行为

学、人类信息化行为学和代理技术理论是符合时代发展需要的。

计算机科学的理论研究应当始终跟随信息化的发展而发展。计算机科学主要是站在计算机技术供应商的视角研究问题的，而现在需要站在信息化的应用者角度去研究新的、与计算机科学有紧密关系的新科学：信息化科学。

1.2 现代数学与计算机科学

早期的数学依赖于物理和自然，是物理和自然的忠实抽象。从历史的长河来看，数学是唯物产生的；但从一个片段来看，数学又是唯心的，而且表现出相当大的主观能动性。抽象化的发展使得数学变成了一堆结构，而每一个结构都建立在自己的公理体系上，而这些公理是任意的、人造的，不再是包含在它里面的概念的真理。这时的数学是纯粹数学。数学的分支非常多，这些数学分支早期建立在 Euclid 几何的相容性上。许多数学分支的提出是由于数学中的悖论引起的，例如 Conter 悖论、Russell 悖论和 Gödel 的不完备性定理等。包含着通常逻辑和数论的一个系统无矛盾是不可能的。在数学家对数学的真谛进行反思和辩论的过程中，产生了直觉数学或构造主义数学学派。直觉数学学派是一个独立的哲学派别，也是一种实体数学，其基本观点为：

- ◆ 不承认任何先验的逻辑原则和所谓具有真值意义的基本公理。
- ◆ 直觉决定概念的正确与否，直觉超过数学的证明。
- ◆ 元数学概念是直觉正确的。
- ◆ 定义和证明是有限构造的、过程性和算法性的。证明如果是非构造的，则是不能接受的；构造如果是无限的，

那么也是不能接受的。

- ◆ 否认排中律和反证法的正确性。反证法是不能接受的。
- ◆ 直觉主义数学认为直觉超过数学的证明，提出自然演绎系统。

直觉数学派也有局部上的分歧，因此产生了不同的学派。例如，现代直觉数学学派大致划分为：Russian 构造主义、Bishop 构造主义、递归分析构造主义、客观直觉主义、Brouwerian 直觉主义和 Martin_Löf 类型理论等。

直觉数学仅承认构造证明，那么什么是构造证明呢？为什么不承认非构造性证明呢？为了弄清楚这些问题，让我们首先了解一下直觉数学的证明观。

在经典逻辑学中，逻辑常量是真或假两个值，逻辑操作和逻辑函数是在真值域中讨论的。例如真值逻辑 $A \vee B$ ，其含义是：A 真或 B 真，则 $A \vee B$ 真。在直觉数学的逻辑概念中，也有同样的逻辑操作，但是其含义却不相同。直觉逻辑 $A \vee B$ 的含义是： $A \vee B$ 的证明要么是 A 的证明，要么是 B 的证明。在直觉数学的逻辑中没有先验的真或假。

又例如 $(\exists X \in A)P(X)$ ，在经典逻辑学中，它表示在 A 中存在着某个 X 使得断言 P(X) 为真。在直觉数学中没有先验的真或假，除了说展示一个 X 使 P(X) 外，为判断它，我们还能够说什么呢？在直觉数学中， $(\exists X \in A)P(X)$ 表示存在着一种方法，找到一个 a 和 P(a) 的证明。因此，直觉数学有如下的宗旨：

对于所有有意义的断言 A，存在着一个 A 的证明 $p: A \leftrightarrow \exists p(p \text{ is a proof of } A)$ 。

现代数学是计算机科学的基础，这些数学包括数论、逻辑学、集合论（包括模糊集合）、代数理论、函数论、范畴论、拓扑学、

几何学等。由于这些数学分支应用于计算机科学，反过来对这些学科也产生了推进作用。例如，产生了现代计算机的信息论、图形学、图像学、计算机语言学（形式语言学）以及建立在集合论基础上的图论和组合数学等学科。另外，对数学中的构造主义数学学派的发展提供了实践的平台与基础。

计算机科学的发展离不开数学的积极介入，同样，信息化科学的发展也离不开数学的参与，但是，令人遗憾的是，数学家对信息化发展非常不适应（有些数学家对软件的错误尤其不能理解）。信息化的发展需要信息化应用的形式化，这种信息化应用的形式化反过来必然促进基础科学的进步与发展。以逻辑学为例，如果始终把逻辑学作为一种科学基础来研究，而不去开拓逻辑学的应用，那么逻辑学的自身发展一定会受到限制。开拓逻辑学的应用，不能仅仅停留在应用的逻辑解释之上，而应该开拓更加贴近应用的更高层的应用逻辑体系和逻辑概念。其实，时态逻辑、模态逻辑和多值逻辑就是在基础逻辑理论上的开拓，只是这种开拓对发展的要求来说，显得不够。

下面，笔者想对数学家谈一谈软件、系统和信息化中的错误问题。我们在本书的《概述》中谈到过数学科学与信息化应用科学的重要差别，其中最重要的差别是：对于数学而言，证明在形式系统之外，从本质上讲是人类的逻辑推理的社会活动。数学家对错误是最敏感的，不能容忍错误的存在。但是对于信息化应用科学来说，由于人类通常处于信息化应用系统之内，其验证活动是人类利用信息化活动本身，而不是逻辑的推理证明进行的。显然，信息化应用科学与数学是不相同的。由于人类成为应用系统的组成部分，所以信息化应用科学把系统中存在错误几乎看成“公理”和“前提条件”。信息化应用科学的第一要务是应用，而不是正确或完美。在应用中逐步去修正错误，在信息化产品的升级

换代中去求得成本允许的正确与完美，而正确与完美反过来为信息化应用降低成本和启发新的发展思路，以产生完美的价值。因此，信息化科学与计算机科学不同，它要研究人类信息化的行为和网络世界中的行为，因为这种行为是与信息化的应用直接相关的。这种行为学的数学与形式化描述，也是信息化理论的要求，而且是信息化逐步走向高级阶段的必然要求。因此，数学与信息化应用科学都有自己的价值观念，都应当得到尊重。

请读者或关注信息化的数学家注意，不能把这种行为理论研究理解为软件执行的输入/输出条件的满足，因为这种软件执行的输入/输出条件的满足是整个行为概念中非常小的部分，以为正确性是解决软件问题的基础。20 多年前，笔者在给学生讲授程序的部分正确和完全正确概念及 E. W. Dijkstra 提出的程序最弱前置条件公理语义方法时也是这么认为的。长期的信息化与总体工作的实践与研究，使笔者逐步认识到，这种理论不能推广的原因恰恰是这个输入/输出条件提出的本身存在着问题。那么，从什么角度提出软件的输入/输出条件呢？可以从软件功能角度提出（正如软件正确性理论要求的那样），从软件可靠性角度提出，从软件安全性角度提出，从软件的可配置和可管理角度提出，从系统集成角度提出，也可以从融入更大系统的一致性和互操作性角度提出。一句话，这种条件是从不同的关注者的不同关注点提出的，绝不是仅仅根据功能描述的输入/输出条件可以判断的。这样的问题是信息化体系结构研究的任务，而不是软件层面上研究的任务。体系结构的核心概念范畴是人类信息化行为和网络世界行为，直接关系到信息化的需求研究。笔者提醒那些关注信息化科学的工作者，尤其是那些年轻的学者，一定要了解信息化，然后才能去研究信息化科学。笔者经常对自己的学生说：懂得计算机与软件，才有进步的坚实基础；了解信息化中存在的问题，才有需要解决

的问题；懂得计算机科学和信息化科学，才有奋斗的崇高目标。

1.3 计算机可读数学

构造主义数学是计算机和软件的数学。对于计算机科学和软件科学具有实际贡献的数学分支包括：自然演绎系统、范畴论、 Λ 演算、组合逻辑、范畴组合逻辑和 Martin_Löf 类型理论。

1. 自然演绎系统

在直觉逻辑的形式系统中，没有先验的真假值，而是建立一种直觉逻辑的自然演绎系统，在这个自然演绎系统中，没有先验的永真公理。由于缺少公理，所以它允许我们在推理的任何阶段引入一个公式作为一个前提假设。如果用形式 $\Gamma: A$ 表示一个推理式，那么 A 是一个公式，是推理的后件（Succedent）； Γ 是公式的有限集合，是推理的前件（Antecedent）。自然演绎系统的规则是结构规则、逻辑规则、引入规则和削去规则。如果习惯了从公理推导逻辑的习惯，即如果采用公理化的直觉逻辑规则，则可以得到一些基本推理公理和推导规则。

2. 范畴论与类别代数范畴

范畴论（Category Theory）在计算机中有着非常广泛的应用，是软件构造、系统设计甚至总体设计的强大数学武器。范畴论是软件设计人员和系统设计人员进行抽象能力训练的最好课程。范畴论研究象元（Object）类、射元（Morphism）类以及射元复合运算的抽象代数，其数学基础是直觉集合论。

$$C = (\text{Obj}C, \text{Mor}C, \text{dom}, \text{cod}, \circ)$$

范畴论研究的内容主要包括射的复合运算的性质、范畴之间

的射（函子）、函子之间的射（自然转换）、伴侣函子和函子的限等概念与理论。范畴论，尤其是类别代数范畴，在软件工程中得到了广泛的应用。类别代数与程序设计语言的 Class 结构是相通的，一个代数是一个三元组： $A=(S, \Sigma, E)$ ，其中， S 是类别名的有限集合， Σ 是操作的有限集合， E 是等式的有限集合。在类别代数范畴（ Σ -同构、 Σ -同态、 Σ -项， Σ -相等）的基础上，提出了抽象数据类型（ADT）概念，对软件的结构设计起到了直接的指导作用。尤其是构造类别代数体系（类别代数的值集是什么，确定值集的构造函数是什么，其他函数必须由构造函数所定义，类别代数的值集中的两个元素是否相等，是由它们的 Σ -项是否相等决定的），对于软件重用与软件自动生成有根本的贡献。

3. Lambda 演算

λ 演算是数学家 Church 于 1941 年作为一个可计算模型提出的，与图灵机是等价的。在数学中常用的函数形式是 $f(x)$ ，其中， f 是一个函数名，而 x 是函数定义域中的一个对象值，表达式 $f(x)$ 表示函数 f 对 x 对象的施用（Application）。如何定义函数 f 是 λ 抽象的任务。例如函数 $f(x)=x+1$ ，在 λ 抽象中用 $\lambda x. x+1$ 表示函数 f 。而函数 f 对 a 施用，可以表示成 $\{\lambda x. x+1\}(a)$ ，从数学的直观可以看到其结果为 $a+1$ 。 λ 演算主要由 λ 抽象、 λ 表达式、 λ 表达式施用、 λ 表达式的范式（Normal Form）、不动点计算函数和演算终止等概念所组成。

λ 表达式 BNF 定义：

```
<λ-exp> ::= <ide> | <number> | (<λ-exp>)
          | <λ-exp> <λ-exp> | λ <bound-variable> . <λ-exp>
<bound-variable> ::= ( ) | <ide> | <ide>, <bound-variable>
```

其中，<ide>和<number>表示 λ 表达式的原子成分。ide 可以

是函数名、对象名和变量名。其中的第三项是括弧，表示作用域。第四项表示施用运算，例如 FX ， $F(XY)$ 和 $FXYZ$ 等。请注意，施用运算是向左结合的，即 $FXY=((FX)Y)$ 。最后一项是抽象运算的定义。抽象运算的一般形式为 $\lambda x.E$ ，其中的 x 表示约束变元， E 是 λ 表达式的体。 E 中的变元如果不是约束变元，则称之为自由变元。

令自由变元的集合为 FV ，而约束变元的集合为 BV ，则有：

- (1) $FV(\text{numbe}) = \phi$
 $FV(\text{ide}) = \{ \text{ide} \}$
 $FV(\lambda - \text{exp1 } \lambda - \text{exp2}) = FV(\lambda - \text{exp1}) \cup FV(\lambda - \text{exp2})$
 $FV(\lambda x.E) = FV(E) \setminus \{x\}$ ，表示在 $FV(E)$ 中去掉 $\{x\}$ 。
 $FV((\lambda - \text{exp})) = FV(\lambda - \text{exp})$
- (2) $BV(\text{numbe}) = \phi$
 $BV(\text{ide}) = \phi$
 $BV(\lambda - \text{exp1 } \lambda - \text{exp2}) = BV(\lambda - \text{exp1}) \cup BV(\lambda - \text{exp2})$
 $BV(\lambda x.E) = BV(E) \cup \{x\}$
 $BV((\lambda - \text{exp})) = BV(\lambda - \text{exp})$

我们来看一个例子， $\lambda y f(\lambda x g(x,y))$ 是一个 λ 表达式，对于 y ，它局部于上述 λ 表达式的体 $f(\lambda x g(x,y))$ ，是一个约束变元。如果把眼光放窄一点，仅看 λ 表达式 $\lambda x g(x,y)$ ，那么 x 是 $g(x,y)$ 的约束变元，而 y 是 $g(x,y)$ 的自由变元。下面，我们讨论 λ 演算的规则。

替换规则定义 对于任意 λ 表达式 N ， M 和任意变量 x ，替换运算 $N[M/x]$ 将 N 中的所有 x 的自由出现用 M 来替换，并有如下的替换规则：

- (1) $x[M/x] = M$
- (2) $a[M/x] = a$ (当 $a \neq x$ 时)
- (3) $(N_1 N_2)[M/x] = (N_1[M/x])(N_2[M/x])$
- (4) $(\lambda x.Y)[M/x] = \lambda x.Y$
- (5) $(\lambda y.Y)[M/x] = (\lambda y.Y)[M/x]$ (当 $y \neq x$ 且 $y \notin M$ 或 $x \notin Y$ 时)

规则 (5) 表明, $(\lambda x.x+y)[y/x] = (\lambda y.y+y)$ 这种替换是不允许的。

施用规则定义 形式为 $(\lambda x.N)M \blacktriangleright N[M/x]$, 称 $(\lambda x.N)M$ 为繁式 (redex), 称 \blacktriangleright 为化简, 该化简关系有自反和传递两个特性。下面举几个例子:

$$\begin{aligned} &\{ \lambda x.xy \} F \blacktriangleright Fy \\ &\{ \lambda x.y \} F \blacktriangleright y \\ &\{ \lambda x. \{ \lambda y.yx \} z \} v \blacktriangleright \{ \lambda y.yv \} z \blacktriangleright zv \\ &\{ \lambda x.xx \} (\lambda x.xx) \blacktriangleright (\lambda x.xx) (\lambda x.xx) \end{aligned}$$

范式规则定义 如果一个 λ 表达式不再含有繁式, 那么称这个 λ 表达式为范式 (normal form)。一个 λ 表达式不一定总有范式, 化简的路径也不是惟一的, 有时有多个化简路径。

Church-Rosser 1 定理 如果 $U \blacktriangleright X$, $U \blacktriangleright Y$, 那么存在着 $X \blacktriangleright Z$, $Y \blacktriangleright Z$, 即如果 U 存在两个范式 X 和 Y , 那么 $X=Y$ 。

Church-Rosser 2 定理 如果 $X=Y$, 那么存在着 Z 使得 $X \blacktriangleright Z$ 和 $Y \blacktriangleright Z$, 即如果 $X=Y$, 并且 Y 是范式, 那么 $X \blacktriangleright Y$ 。

下面, 我们来介绍 λ 演算的不动点概念。

不动点定义

(1) 如果 $F(f) = f$, 那么 f 是 F 的不动点。

(2) 如果对于任意 F , $\phi(F)$ 是一个不动点, 即有 $F(\phi(F)) = \phi(F)$, 那么 ϕ 是一个不动点计算函数。

不动点定理

(1) $\theta = \lambda a. \lambda b. b(aab)$, $\theta\theta$ 是不动点计算函数。

(2) $Y = \lambda f. (\lambda g. f(gg)) (\lambda g. f(gg))$, Y 是不动点计算函数。

(3) 如果 ϕ 是不动点计算函数, 令 $G = \lambda a. \lambda b. b(ab)$, 那么 $\phi(G)$ 也是一个不动点计算函数。

(4) 存在着无穷多不动点计算函数。

4. 组合逻辑

组合逻辑 (Combinatory Logic) 是由 H. B. Curry 等于 1958 年提出并完善的。组合逻辑是研究 λ 演算和函数施用的逻辑。组合逻辑是一种形式系统, 自 Euclid 初等几何以来, 已经有了许多形式系统, 除了研究的对象和内容与其他形式系统不同外, 单从形式系统的建立方法来看, 组合逻辑也与其他形式系统有所不同。Curry 的组合逻辑学在以下两个方面有明显的特点, 并区别于其他形式系统:

- ◆ 是完全显式的和完全构造的形式系统。
- ◆ 组合逻辑是一个完全施用的形式系统。施用的英文是 Application, 意思是一个函数 f 作用到一个对象上。组合逻辑告诉我们, 一个形式系统最后的操作是“施用”。那些具有范畴或者类型概念的操作, 例如 $+$, $-$, \times , $/$, and, or, not, push 和 top 等都纳入了形式系统 Obs 的集合概念中, 甚至连“等式”这个概念都有类型或范畴的意义, 也都纳入到了形式系统 Obs 的集合中。这个形式系统中只有一个操作, 即“施用”, 简写为 App。在组合逻辑中, 与“施用”构成逆运算的是“抽象” (Abstraction), 组合逻辑的抽象是无变量的抽象, 被抽象的函数是变量的, 最后的表达式仅仅由组合子和函数符号所组成。

所有对象在“施用”的意义上一致 (无类型)。另外, “判定”本身 (证明) 不属于对象概念。组合逻辑中定义了一系列组合子: I, C, W, B, K, S, Φ 和 Ψ 等。它们的定义分别为:

等式组合子 $I \equiv \lambda x.x$

$Ix = x$

交换组合子	$C = \lambda fxy.fyx$	$Cfxy = fyx$
二次组合子	$W = \lambda fx.fxx$	$Wfx = fxx$
复合组合子	$B = \lambda fgx.f(gx)$	$Bfgx = f(gx)$
消去组合子	$K = \lambda fx.f$	$Kcx = c$
分配组合子	$S = \lambda fgx.fx(gx)$	$Sfgx = fx(gx)$
分配组合子	$\Phi = \lambda fghx.f(gx)(hx)$	$\Phi fabx = f(ax)(bx)$
分配组合子	$\Psi = \lambda fgxy.f(gx)(gx)$	$\Psi fgxy = f(gx)(gy)$

其中，S 和 K 组合子是基本的，无形式变量抽象，例如如下的公式：

$$\lambda x.(x+1)^2$$

可以转换成 B(W(mult))(C(plus) I)组合子的表达式。

这种抽象是具有实际意义的，例如函数 $f(x.g(y.h(z)))$ 。函数施用与变量的表达式是函数名与变量混合在一起，我们希望能够找到一种转换，构造一个新的函数表示形式 $F: \langle x,y,z \rangle$ 。

5 . 范畴组合逻辑

范畴组合逻辑既有范畴论，又有组合逻辑。组合是指 Curry 的组合逻辑学。在这里，研究范畴组合逻辑是借鉴组合逻辑的思想，将范畴论的形式系统也依照组合逻辑的方法构建，在形式定义中只有射元和组合子，去掉象元进行组合。范畴论的组合逻辑的最后操作是什么呢？研究范畴组合逻辑的人认为是“复合”而不是“施用”，范畴组合逻辑把在组合逻辑中定义的最后的“施用”操作也当做射元，在复合的意义上研究所有射元组合的规律，这种研究范畴论的组合逻辑学称为范畴组合逻辑。

在范畴组合逻辑中，范畴的定义形式也做了改变，范畴为如下的二元素： $C = (M, \circ)$ ，其中的 M 即为 $MorC$ ，而“ \circ ”表示复合运算。但是，在这里把“ \circ ”定义为部分操作。此时，“ \circ ”满足如下条件：

(1) 如果 $f \circ g$ 和 $h \circ f$ 有定义, 那么 $(f \circ g) \circ h$ 和 $f \circ (g \circ h)$ 是有定义的。

(2) $f, g, h \in M$, 当且仅当 $f \circ (g \circ h)$ 有定义且 $(f \circ g) \circ h = f \circ (g \circ h)$ 时, $(f \circ g) \circ h$ 有定义。

(3) 存在一个单位射元, $1_A, 1_A \in \text{Mor} \mathbf{C}$ 使得:

$$f \circ 1_B = f$$

$$1_A \circ f = f$$

例如, 一个小范畴的形式系统可以描述为:

Primitive Ideas

有如下的对象概念:

◆ $A, B, C, \dots, f, g, h, \dots, 1_A, 1_B, 1_C, \dots, \varepsilon, \neg, \perp,$

\perp

◆ dom, cod

◆ $(-, -), \text{Fst}, \text{Snd}, [-, -], \text{Fst}^*, \text{Snd}^*, (-) \bullet (-), \text{head}, \text{tail},$
 $(-) \rightarrow (-), \text{app}, \text{cur}.$

而范畴的操作仅仅为:

◆ 一个二元复合操作 “ \circ ”。

◆ 如果 $\text{dom}(f), \text{cod}(g)$ 是 obs, 并且有 $\text{dom}(f) = \text{cod}(g)$, 那么 $f \circ g$ 也是 obs。

给出一个二元谓词 (=) 和基本的等式规则假定:

$$(\rho) f = f$$

$$(\text{dom}) \text{ if } f = g \text{ then } \text{dom}(f) = \text{dom}(g)$$

$$(\text{cod}) \text{ if } f = g \text{ then } \text{cod}(f) = \text{cod}(g)$$

$$(\text{compaq}) \text{ if } f = g \text{ and } h = k \text{ then } f \circ k = f \circ h$$

$$(\text{domc}) \text{ dom}(f \circ g) = \text{dom}(g)$$

- (codc) $\text{cod}(f \circ g) = \text{cod}(f)$
 (assc) $(f \circ g) \circ h = f \circ (g \circ h)$
 (Idr) if $\text{dom}(f) = A$ then $f \circ 1_A = f$
 (Idl) if $\text{cod}(f) = B$ then $1_B \circ f = f$

有如下的特性:

(-, -)特性

- (pair) $(f, g) \circ h = (f \circ h, g \circ h)$
 (Fst) $\text{Fst} \circ (f, g) = f$
 (Snd) $\text{Snd} \circ (f, g) = g$
 (paireq) if $f = g$ and $h = k$ then $(f, k) = (g, h)$
 (\top) if $\text{cod}(f) = \top$ then $f: A \rightarrow \top$
 (pairde) $h = (\text{Fst} \circ h, \text{Snd} \circ h)$

[-, -]特性

- (union) $[f, g] \circ h = [f \circ h, g \circ h]$
 (Fst*) $[f, g] \circ \text{Fst}^* = f$
 (Snd*) $[f, g] \circ \text{Snd}^* = g$
 (unioneq) if $f = g$ and $h = k$ then $[f, h] = [f, k]$
 (\perp) if $\text{dom}(f) = \perp$ then $f: \perp \rightarrow A$
 (unionde) $h = [h \circ \text{Fst}^*, h \circ \text{Snd}^*]$

(-)•(-)特性

- (concat) $(f \bullet g) \circ (h \bullet k) = (f \circ h) \bullet (g \circ k)$
 (ε Idr) $f \bullet \varepsilon = f$
 (ε Idl) $\varepsilon \bullet f = f$
 (assmon) $(f \bullet g) \bullet h = f \bullet (g \bullet h)$
 (concateq) if $f = g$ and $h = k$ then $f \bullet h = f \bullet k$
 (head) $\text{head} \circ (f \bullet g) = f$
 (tail) $\text{tail} \circ (f \bullet g) = g$

$$(\text{concatde})\ h = (\text{head} \circ h) \bullet (\text{tail} \circ h)$$

$(-)\rightarrow(-)$ 特性

$$(\rightarrow) \quad \lambda f.b(f) \circ \lambda g.b(g) = \lambda(f \circ g).b(f \circ g)$$

$$(\text{cur}) \quad \text{cur}(f) \circ g \circ h = f \circ (g \circ h)$$

$$(\text{app}) \quad \text{app} \circ (f, g) = f \circ g$$

$$(\rightarrow d1) \quad h = \text{cur}(\text{app} \circ \langle h \circ \text{Fst}, \text{Snd} \rangle)$$

$$(\rightarrow d2) \quad f = \text{app} \circ (\text{cur}(f) \circ \text{Fst}, \text{Snd})$$

$$(\rightarrow \text{eq}) \quad \text{if } f = g \text{ then } \text{cur}(f) = \text{cur}(g)$$

$$(\rightarrow \text{eq}) \quad \text{cur}(\text{app}) = 1$$

6. Martin_Löf 的类型理论

Martin_Löf 分别在 1975 年、1979 年和 1990 年提出和完善了其类型理论（构造数学）体系。Martin_Löf 说研究类型论以数学逻辑作为数学基础，他认为没有一个对象是没有类型的，它不仅存在于数值、操作中，而且还存在于证明之中。Martin_Löf 类型理论的意义还在于建立了一种新的形式系统，这个新的形式系统至少有两个特点：第一，建立一种新的哲学与逻辑意义上的集合与范畴论系统，以解决一些数学的基础问题；第二，现有的数学符号体系与语言不适合计算机或程序设计语言。例如，一阶谓词演算系统语言不适合程序设计语言，这也是计算机科学要发展自己的程序设计语言的原因。用通俗的话说，就是要研究适合机器读的数学语言，至今为止，所有的数学语言都是人读的语言。Martin_Löf 认为类型理论更适合程序设计和计算机科学。Martin_Löf 类型理论的哲学观点是：

- ◆ 世界是按类型构造的（包括思维和表示都是类型的）。
- ◆ 类型是层次的。
- ◆ 低层次类型是高层次类型的成员。

类型理论的数学观点是：把类型论作为数学的基础来研究，立足在范畴论的高度上。同时，Martin_Löf 还批判组合逻辑和范畴论的无类型，为它们给出类型。他强调没有一个数学对象是不属于某个类型的，包括数值、函数、操作和证明。虽然没有先验的真值，但他把真值和证明联系起来，真为存在证明，假为不存在证明。将证明纳入到形式系统之内，与其他直觉数学一样，也引入了构造规则、引入规则、消去规则和等式规则。

但是，在 Martin_Löf 类型理论的早期讨论中，并没有对“无穷对象”进行构造。可以说，所有对象都是构造的，而无穷在那时的类型理论中只是一个概念，而不是对象。在定义如下类型世界时：

$$V_0 \in V_1 \in V_2 \in \dots$$

把 V_i 视为 V_{i+1} 的对象， V_i 是无穷的，但在类型理论中，并没有给出无穷作为对象的讨论。1990 年，Martin_Löf 在文献《Mathematics of Infinity》中，对他的无穷数学做了补充说明，完成了无穷对象的构造，并讨论了无穷的层次概念。

希望读者理解这种无穷对象的构造，就是对停机问题的判定。如果能够把无穷判定为一个对象，那么就是表示对无穷计算机的停机问题做出判断。

Martin_Löf 类型理论中的逻辑运算符号的含义如表 1.1 所示。

表 1.1 类型理论中的命题证明及其含义

命题证明	含义
\perp	没有任何东西可以作为 \perp 的证明
$A \& B$	(a, b) ，其中， a 是 A 的一个证明， b 是 B 的一个证明
$A \vee B$	$i(a)$ or $j(b)$ ，其中， a 是 A 的一个证明， b 是 B 的一个证明

(续表)

命题证明	含义
$A \supset B$	$\lambda x.b(x)$, 其中, 如果 a 是 A 的一个证明, 则 $b(a)$ 是 B 的一个证明
$(\forall x)B(X)$	$\lambda x.b(x)$, 其中, 如果 a 是一个个体, 则 $b(a)$ 是 B 的一个证明
$(\exists x)B(X)$	$(a.b)$, 其中, a 是一个个体, b 是 $B(a)$ 的一个证明

组合逻辑和范畴组合逻辑认为, 存在着超越所有类型的操作 (或者说最后的操作), 即“施用”或“复合”。所谓“超越所有类型”, 就是说它适用于一切类型, “施用”或“复合”操作在一切类型中被直接使用着。将这些概念与 Gödel 的不完备性定理中的包含数论的任何系统都必定存在着不可判定的命题, 或者数论中存在一个命题, 它是真的但是是不可证明的等概念比较, 也许读者会有一些想法。

谈到这里, 我们不能不谈谈人工智能科学的发展情况。

在 20 世纪 80 年代, 人工智能技术曾经热闹一时, 除个别内容的研究有一定的进展外, 热闹之后又走向了研究的低谷。在这期间, 最为有名的研究计划是日本的第五代计算机计划。当时, 日本已经在芯片技术研究的很多方面接近或超过美国, 但是在软件方面还受制于美国, 在操作系统方面更是如此。该计划是研究新一代计算机, 不用美国的操作系统, 实现跳跃发展。在英国人的帮助下, 该计划采用 PROLOG 语言, 企图跳过这代计算机, 从根本上超过美国。结果, 计划失败了, 失败的原因不是因为跳跃式发展, 而是对技术发展没有把握好。很有意思的是, 当时我国在人工智能研究方面没有走得太远, 这与我国的计算机科学家比较清醒有关。

虚拟现实是在可视化技术发展过程中提出的, 可视化的成功

促成了对计算机的触觉、视觉、听觉和其他感觉的研究，加强了对人机交互的新阶段的研究，企图对人工智能实施进一步的冲击，把人工智能的发展进程演变为：

计算→理解（可视化）→虚拟现实→人工智能化更高阶段

虚拟现实的研究同样遇到了现代电子计算机能力的约束，造成了体制上的困难。

人工智能研究中曾经提出过许多概念，例如模式识别、自然语言理解、符号系统、机器推理与证明、知识工程、机器人工程和智能代理等。人工智能研究的战场在两个方面：一个方面是以机器人为主要的研究领域，主要从硬件和软件的角度进行研究；另一个方面是网络世界中以知识工程、智能代理为中心的研究领域，主要从软件和网络角度进行研究。

笔者在研究类型程序设计方法支持软件智能化时，对人工智能的一些问题进行了研究。笔者对软件智能化的看法是：软件智能化不是通过知识库和规则库等方法实现的，而必须通过软件自身的逻辑能力、归纳能力、证明能力和分析能力等来实现。软件智能化必须将上述能力形式化和可执行化。软件构造和构造数学是软件智能化的原理与基础。笔者还认为：软件智能化必须对计算机科学最基础的概念——“计算”或“替换”（也可以称做化约或规约）进行区别，软件的逻辑能力、归纳能力、证明能力和分析能力等应当是通过符号的处理逻辑能力达到的。例如，在有限集合中，对于 $(\forall X \in A)P(X)$ ，可以采用数值计算的方法来完成；而对于可枚举无穷集合，采用数值计算方法不能停机。使计算方法具有智能化的方法为归纳法，通过证明的方法，结论的得出是可以终止的。

人类的智慧成果千千万万，但是最能代表人类智慧的科学是数学，而集中反映人类智慧能力的是“实数理论”。实数理论集

中地反映了现代人类的宇宙观，微观到无限小，表现在数轴上一个没有体积的点上，可以无限密集，即把无限小概念进一步划分为：一阶无限小，二阶无限小，三阶无限小，直到无限阶的无限小。从宏观上看，达到无限大，表现在数轴的无限长度之上，而且还可以把无限大概念进一步划分为：一阶无限大，二阶无限大，三阶无限大，直到无限阶的无限大。在上述的无限理论中，依然存在着可以想像的空间。在无限空间中开辟多维空间，把无限的时间与无限的空间构成更加难以琢磨的新体系。人类的思考能力具有无限的不确定性，为人类的创新奠定了最坚实、最广泛的基础。当然，人类个体是不具有全部人类的智慧与能力的，人类的智慧是以人类的整个群体表现出来的。

研究人工智能科学，必须从个体、组织群体和整个人类群体三个层面来进行，它们有如下的关系：

人类个体能力<人类具体组织群体能力<整个人类群体能力

而过去的人工智能科学主要研究个体的人工智能问题，显然，对人类具体组织群体和整个人类群体的智能问题的研究是很不够的，不仅应该从典型的现代人的聪明个体出发，而且还应当从人类整体的群体资源出发，来研究人工智能科学。

人类科学或者人类行为科学的研究是建立在具有如此智慧的科学基础之上的，其智能能力具有最大的无限性、最快的思维速度、最大的不确定性和最大的群体资源能力。

而现代计算机科学是建立在递归可枚举集合理论基础上的，说得不好听些，是建立在自然数基础上的（从数学的角度看，递归可枚举集合与自然数还是不同的）。人类科学与计算机科学在能力上的差别表现在实数理论和递归可枚举集合上。任何一个学习过高等数学的人都知道实数理论和递归可枚举集合有多大的差别，从它们的表现能力上，也可以看出它们的差别有多大。也就

是说，计算机科学为计算机表现出的能力远远小于数学为人类表现出的能力，它们之间的差别无限大。笔者认为，当代的人工智能水平相当于“自然数”水平，还有很长的路要走。

1.4 信息化与形式化

讨论信息化与形式化的目的是为网络世界建立完整的数学体系与哲学体系。从前面介绍的计算机可读数学中可以看出，仅仅这些数学，对于建立网络世界中的完善的数学体系是不够的，我们需要更多的网络世界的数学。我们还需要计算机和网络世界的哲学体系，只有真正的网络世界的哲学体系建立起来，哪怕是初步的，才标志着网络世界有真正含义上的人工智能。

信息化应用需要形式化，形式化可以加速信息化应用的进步，降低信息化应用的成本与风险。信息化的应用如此之多，难道要为各行业的业务实行形式化吗？面对这些问题，由于实践与经历，早期的计算机科学家感到最难于介入的就是信息化了，他们往往面对庞杂的信息化实践一筹莫展。新型的信息化科学家与理论学家必须对信息化有所认识，至少应当有兴趣与信息化工作者合作，了解信息化需要解决的问题。

如何对应用进行形式化呢？

通过几十年的信息化，人们对信息化应用的认识也逐步深刻起来。

对信息化解决方案的研究使我们逐步认识到，信息化体系结构或总体学研究是问题的核心。在实践中认识到并产生了共识，即设计信息系统总体结构，应当从分三个范畴入手，这三个范畴是：运营体系结构、系统体系结构与技术体系结构。现在的问题

是，应当产生描述这三个范畴的理论方法，使这种体系结构的理论方法能够有效地指导信息化的实践，并能够从理论上覆盖信息化的全局活动。

体系结构研究必须建立在坚实的理论研究基础之上。这个用来指导体系结构研究的理论实际上是人类世界（人类活动的世界）和网络世界（网络系统中主体活动的世界）两个行为学理论体系及它们的结合。

人类世界的人类行为研究包括对个体和群体的各种行为研究，例如生物圈行为、家庭行为、宗教行为、社会行为、文化行为、政治行为、军事行为、经济行为、生产行为、消费行为、学习行为、科学行为、企业行为、团体行为和国家行为等。人类的许多行为都是在理性指导下，认识环境、改造环境并且改造自己的。人类的许多行为是可以研究其目的、动机、意念和心理等主观特性的。研究人类行为，包括研究社会行为和组织行为。所谓组织的业务行为，是指组织和组织中的成员为了业务目的所从事的活动或行为，例如政治行为、经济行为、商业行为、生产行为和社会行为等。我们称这些行为为“组织业务行为”或“组织功能行为”。

由于网络世界中的主体从本质上说是软件运行的进程，所以网络世界中的行为研究也可以说是软件行为的研究。软件行为学实际上是研究行为的语义学，这种行为的语义描述是计算机可读的，而不仅是人类可读的。软件的行为也是系统行为。

信息化体系结构的研究主要是研究人类世界中的主体使用信息技术与产品的行为，这些行为包括维系组织关系的“组织协同行为”和体现组织存在价值的“组织业务行为”，这两种行为汇合在一起构成运营的行为。研究组织运营行为的结构关系描述称为运营体系结构。针对运营体系结构要求进行信息化的建设，建

立一些信息系统，这些系统在网络世界中同样通过行为来表现系统的功能和能力，这样的系统描述称为系统体系结构。建立系统体系结构与运营体系结构的行为的一致性描述，是信息化应用形式化的根本目标。

通过上述分析，信息化的形式化研究对象与目标就明确了。把行为与行为的结果（信息的内容）作为形式化的对象是新时期计算机领域或信息化领域的主要工作内容。正因为如此，在本书中，我们列举了代理化、信息化总体学、体系结构、互操作性理论、安全性理论、服务性理论、信息化监管理论、信息化认证理论、网络对抗理论、群体计算理论、网络世界行为学、人类信息化行为、信息化技术法规、信息化标准化理论、信息化测评认证、社会科学形式化和哲学形式化等系列问题，对这些问题的研究都可以采用形式化方法。

本书中介绍的人类世界中人类行为（活动）学的研究、人类世界人类信息化行为（活动）的研究和网络世界中虚拟主体（代理）行为学的研究，可以说开创了以行为学为中心的形式化的研究工作，它已经不是自动机、计算机、程序语言和系统之类概念的形式化定义的研究工作。人类世界中的人类行为（活动）和人类信息化行为（活动）两个方面的研究开辟了可信息化社会科学的形式化的研究工作。下面，将笔者在《软件行为学》一书中的形式化体会介绍给各位读者，供读者参考。

在《软件行为学》中，为什么要选择范畴论作为研究信息化的数学基础呢？对于多主体多行为描述，主要是讨论多个行为的关系和行为之间的运算。为了描述多个行为的关系与运算，定义了一个行为表达式或者行为表达式语言。定义行为表达式首先要搞清楚行为之间的关系。例如，我们把“主体对客体的操作”定义为一种原子行为。进一步思考，我们可以用行为之间的二元操

作或运算定义行为的基本运算，例如行为连接、顺序、平行、单向条件、双向行为和选择行为等。我们还可以将行为定义到更高层次，把“主体对行为的操作”定义为“服务”的概念，这实际上是一种复合行为。

我们在讨论网络世界的行为时，总喜欢选择范畴论作为研究信息化的数学基础。例如，可以把基本行为概念与范畴论中的“射元”（morphism）类比，而把行为应用模式中更高层的行为概念（例如服务、管理、监管、控制、认证和对抗等）理解为“函子”。如果再把函子看做一个范畴，那么函子的射便是自然转换，这样，便可研究伴侣函子和函子的限，甚至可以继续研究范畴的范畴等理论问题。对于多行为的描述性的语义或形式定义，我们采用行为表达式。这个行为表达式既可以描述原子行为（主体对客体的操作）、二元关系行为和复合行为（主体对行为的操作），又可以在此基础上进一步研究行为范畴的更高层的概念。例如，对行为表达式的本体语义、逻辑语义、态势语义、安全语义和类型指称语义进行描述，并把这样的方法反复运用到各种行为应用模式中。

例如，在《软件行为学》中，我们做了如下的形式规定：

在人类世界中，首先可以定义一些人类的原子行为的形式描述方法，例如：

$\alpha = s: a()$	-----	非及物行为
$\alpha = s: a(?)$	-----	不确定及物行为
$\alpha = s: a(O)$	-----	确定及物行为
$\alpha = \alpha_1 \alpha_2 \dots \alpha_n$	-----	行为踪迹

在网络世界中，首先可以定义一些虚拟主体的原子行为的形式描述方法，例如：

$\alpha = S: f \rightarrow O$	-----	申请行为
-------------------------------	-------	------

$\alpha = S: f(?)$ ----- 不确定执行行为
 $\alpha = S: f(O)$ ----- 确定执行行为
 $\alpha = \alpha_1\alpha_2 \dots \alpha_n$ ----- 行为踪迹

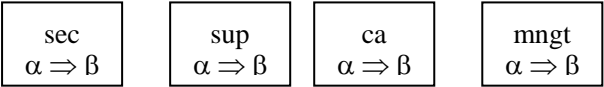
在原子行为形式定义的基础上，我们还可以定义一些常用的或者我们关注的复合行为，例如：

$S[O]$ ----- 客体的伴侣行为，也作为原子行为
 $S_1(O) \Rightarrow S_2$ ----- 交易类的单向行为
 $S_1(O_1) \Leftrightarrow S_2(O_2)$ ----- 协同类的双向行为
 $S: F(\alpha)$ ----- 主体通过函子 F 对行为进行操作
 $S_2 < S_1 >: F(\alpha_1)$ ----- 主体 S_2 通过函子 F 对所伴主体 S_1 的行为 α_1 进行操作

对行为进行运算，对单一行为或在两个行为之间建立某种操作关系，例如：

$\alpha; \alpha$ ----- 行为的顺序操作，表示行为顺序发生
 $\alpha_1\alpha_2$ ----- 行为连接操作，表示行为的踪迹
 $\alpha \Rightarrow \alpha$ ----- 两个行为条件单向操作
 $\alpha \Leftrightarrow \alpha$ ----- 行为对等双向操作
 $\alpha \parallel \alpha$ ----- 平行行为（不确定时间顺序的多行为）
 $\alpha' | ' \alpha$ ----- 在两个行为之间进行选择
 (α) ----- 行为的括弧操作，表示某种层次关系
 $A < B >$ ----- 主体伴侣操作
 $S[O]$ ----- 客体伴侣操作

有了上述形式化的基本定义形式，我们就可以得到行为模式定义，例如，条件单向行为 $\alpha \Rightarrow \beta$ 应用到行为控制模式、行为监管模式、行为认证模式和行为管理模式上，可以用如下的符号表示：



对于双向对等行为 $\alpha \Leftrightarrow \beta$, 可以用如下方式表示行为协同、行为对抗、交易、交换和互操作等模式。有时, 我们还可以用 $\alpha \parallel \beta$ (平行行为) 来表示行为对抗形式或者同时活动的模式。

ass $\alpha \Leftrightarrow \beta$	count $\alpha \Leftrightarrow \beta$	trans $\alpha \Leftrightarrow \beta$	exch $\alpha \Leftrightarrow \beta$	interop $\alpha \Leftrightarrow \beta$
---------------------------------------	---	---	--	---

我们可以借鉴数学中的抽象方法 (例如 Lambda 抽象) 对行为实行抽象, 例如, 可以得到:

- ◆ $\lambda(\alpha).(S: F(\alpha))$
- ◆ $\lambda(F).(S: F(\alpha))$
- ◆ $\lambda(\alpha, F).(S_2 < S_1 >: F(\alpha))$

有了这些行为的函数、范畴论的函子和 Lambda 演算等形式, 我们还可以对行为的应用行为本体语义和逻辑问题进行研究。

另外, 我们还可以对行为的主体和客体进行形式规定, 例如:

```

S ::= man | women | user | process | subprocess | agent | MultiAgents
    | Organization | S[O] | S2 < S1 > ----- 主体
O ::= object | component | program | file | e_mail | ... | S[O]
                                ----- 客体

```

我们可以对行为主体的组织结构关系, 例如代理、代理集合、代理网络、多代理系统、代理平台和代理网格, 给出如下的形式化定义:

```

ag =  $\lambda$ (ma, mapf, vs) • Frame(ma, mapf, vs)
mapf =  $\lambda$ (ma) • AgentEnv(ma, io, pb, ps)
ma =  $\lambda$ (an) • Organization(an, mas, mb, ms)
an : AgentNet = AgentSet × Channels
as : AgentSet = {x | x: Agent }

```

```

ch: Channels = {(x•α, y•β) | x, y: Agent and x•α is a port of the
                agent x and y•β is a port of the agent y}
a: Agent

```

其中，

```

Frame: MultiAgents×AgentPlatform× VisualizationSystem→
      AgentGrid
AgentEnv: MultiAgents×Interoperability×Business×State→
      AgentGridPlatform
Organization: AgentNet ×Association×Business×State→
      MultiAgent

```

是一些函子。

上述介绍就是笔者在《软件行为学》一书中建立形式化的思路与原则。这项工作没有前人的成果可以参考，或者说可借鉴的东西很少，完全是根据事物的要求，在自己长年的计算机科学研究基础上一点一点积累起来的（历时 8 年）。当然，读者也许有更好的形式化方法。

为信息化进行形式化不仅仅是为了好看，也不仅仅是为了解释行为学的问题，更重要的是为了使这个形式化体系对计算机可读，甚至将这种形式化模型融入到计算机、网络、软件和系统之中，使之真正具有相当的智能化能力。

在形式化的研究中，唐稚松、周巢尘、李未、陈火旺和陆汝钊等许多著名学者分别在时态逻辑、语义学数学、形式化方法理论、实时系统形式设计的时段演算方法、分布式程序设计方法、操作语义学理论、程序设计语言理论和知识工程等方面做出了各自的贡献。另外，近些年来，陈钟、梅宏、陈向群、张世琨、王怀民、毛新军、李舟军、毛晓光、王戟、李军、史元春、徐晓飞、吕建、怀进鹏、付玉熙、徐宝文、赵文耘、金枝、母国庆、应时

等一大批年轻的计算机科学工作者，也开始关注计算机、软件、网络系统和程序设计的科学与形式化问题。

但是，大多数的形式化理论研究依然停留在计算机或者软件设计等概念上，对于信息化和信息系统形式化描述，几乎只有笔者及其 QNS 工作室、信息安全产业和信息化应用领域的理论工作者在研究。因为笔者长期参与和主持了许多国家信息化工程的总体工作，所以更加体会到了什么是国家信息化急需解决的问题，什么是信息化主战场的理论工作。信息化中的系统集成、互操作性、网络服务、系统安全、网络犯罪、网络监管、网络认证、网络对抗、网络群体计算、信息化综合效益、体系结构、用户标准化、技术法规、测评认证和系统评估等重大理论问题急需有人研究。尽管很少有前人研究过这些理论问题，也会有较大风险，但我们面对这些问题，也不能不管不顾，而应该安下心来去研究这些非常急迫的理论问题。多年的信息化经验教训告诉笔者，信息化最需要抓住机遇，不要回头看。总之，我们期盼有更多的理论工作者参与到我国信息化主战场的理论研究中来。

第 2 章

计 算 理 论

2.1 可计算性理论

计算机科学的首要理论是计算模型理论。例如，一阶谓词演算模型、图灵机模型、冯·诺依曼计算机模型、递归函数论模型（不动点理论）、代数模型、Lambda 演算、组合逻辑、范畴论、类别代数、重写理论、形式语言理论、时态逻辑和 Petri 网络等，这些模型是等价的。可计算理论是计算机科学最重要的理论基础。所谓“可计算”，是指图灵机可计算。计算模型理论和可计算理论是进行新一代计算机模型和人工智能研究必须学习的理论，是计算机专业硕士生和博士生的理论课程。可计算理论中最为著名的结论是“图灵机的停机问题不可判定”。对于经典的可计算理论，我们知道，单带图灵机与多带图灵机在可计算方面是等价的。**笔者想问：无穷多带图灵机与单带图灵机在可计算方面还等价吗？**又例如图灵机停机问题，是否可以理解一阶无穷计算机判定一阶无穷计算机的停机问题是不可判定的。但是，如果对于二阶无穷计算机或更高阶无穷计算机判定一阶无穷计算机停机问题是否可以判定？什么是高阶无穷计算机？显然应当有新的计算机模型。请读者注意，研究理论问题需要关注无穷，而做具体工作往往是有限的。对于群体计算网络，尤其对于可移动、可生长的群体多代理系统，可以看成准无限资源计算机系统，关于这方面的内容将在后面的群体计算理论中详细介绍。

先有可计算理论，例如图灵机和冯·诺依曼计算机模型在先，而后有计算机。请注意，上述的许多计算模型都产生过相应的计算机，除当代的随机存储计算机体系结构依据冯·诺依曼计算模型之外，Lambda 演算产生过 LISP 语言，并且产生了相应的 LISP 计算机体系结构；组合逻辑计算模型产生过 FP 程序设计和组合逻辑

计算机；逻辑演算曾经产生过 PROLOG 逻辑程序设计语言和风靡一时的第五代计算机。需要提醒读者注意的是，这些计算机模型没有成为时尚或消失，并不完全是由于模型不好，很大程度上是由应用、市场与发展要求所决定的。

2.2 算法理论

1. 算法复杂性理论

计算机科学的另一个理论基础是算法理论和算法复杂性理论。算法理论是程序设计的必学理论，是软件和计算机应用专业的大学生、硕士生、博士生的学习课程。算法的理论方法是组合数学，本质是枚举算法，这可以从 $\exists x \in A \{ x \mid P(x) \}$, $\forall x \in A \{ x \mid P(x) \}$ 看出。算法复杂性分析理论研究空间复杂性和时间复杂性。算法复杂性可以是线性复杂性 $O(n)$ ，也可以是多项式复杂性 $O(n^c)$ 和指数复杂性 $O(c^n)$ 。一般认为，多项式复杂性是可以忍受的，而指数复杂性有问题，在大指数的情况下（即 n 取较大数值时），计算难以停下来。算法理论中有个非常著名的问题：NP 完全类问题，即指数复杂性是否可以转换成多项式复杂性。算法复杂性的计算模型包括：时间界限图灵机、确定图灵机和不确定图灵机。通常，提高算法效率的方法包括：加权算法、概率算法、模糊算法、并行算法和算法信息论方法。

但是，对于群体计算网络，尤其是可移动、可生长的群体多代理系统，在相当范围内能够实现不确定计算，可以将指数复杂性问题变成低阶多项式复杂性问题。关于这点，在本书的群体计算理论中还会做比较详细的介绍。

2. 算法信息论

在一个封闭的信息世界中，信息是守恒的，它既不能产生，也不能消失。但是，信息的存在形式是可以转化的。有些信息可以被计算机直接感受，而有些信息不能被计算机直接感受。

算法效率的提高在于减少空操作和重复操作，应注意算法信息的执行回收，不要丢弃算法执行中产生的信息。

3. 算法的设计步骤

算法的设计步骤如下：

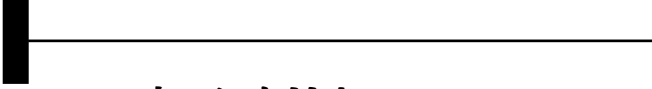
- (1) 算法信息的表示设计。
- (2) 是否考虑算法信息回收。
- (3) 算法步的设计。

◆ 不确定算法步：递归、回溯。

◆ 确定算法步：循环迭代。

- (4) 终止条件的确定。
- (5) 复杂性分析（空间和时间的使用效率）。
- (6) 改进算法，重复第（1）步到第（5）步，直到满意为止。

第 3 章



电子计算机

3.1 单机、多机到网络虚拟计算机

电子计算机硬件经历了电子管、晶体管、集成电路、大规模集成电路和现代的超大规模集成电路几个发展阶段。在体系结构上,计算机从单一 CPU 逐步走向多 CPU;存储器从几千字节发展到千兆字节;从执行一个单道程序发展到执行成百上千个并发进程;从以存储器、CPU 和总线结构为核心发展到以外部磁盘存储器为核心;内部连接结构从单总线发展到开关多总线;从单机走向多计算机构成集群的体系结构;在开发方面,从单一厂商产品发展到走向世界范围的多厂商集成。

无论如何发展,从概念上讲,电子计算机必须从把计算机封装在单个或多个机箱内的物理计算机发展到网络范畴的虚拟计算机。

计算机封装在单个或多个机箱内,可以有几十、几百、几千个 CPU。如果要提高计算机的处理能力,把几万、几十万、几百万、几千万或几亿个 CPU 封装在一个或几十个计算机机箱内,该如何构造呢?

把网络看成计算机是一种重要的概念进步——虚拟的计算机概念。在过去的计算机发展实践中,曾经把几十、几百、几千个计算机组织在一个网络上,构成一个虚拟的计算机进行计算。如果提高网络计算机的处理能力,把一个单个计算机作为处理单元,那么几万、几十万、几百万、几千万、几亿、几十亿个计算机组成的网络如何发挥计算的效力?如何有效地使用、调度计算资源?

为了解决这些问题,计算机的发展需要新的计算模型理论,需要探索新计算体系。显然,我们不能停留在传统的自动机理论、

图灵机理论、递归函数论、冯·诺依曼计算机模型等理论研究上，这些理论在面对上述问题时显得力不从心。

3.2 高阶无穷计算机

我们在上一节中已经谈到了一阶无穷计算机、二阶无穷计算机和高阶无穷计算机。群体计算网络，尤其是可移动、可生长的群体多代理系统，可以看成是一个无限资源计算机系统和不确定计算系统。从虚拟计算机概念考虑，如果把传统的单机计算机或多机计算机看成一阶无穷计算机（其资源可以无限外延）。那么，我们可以把当前全球网络中的所有计算机看成是一个虚拟计算机，这个虚拟计算机在群体计算网络中便是一个二阶无穷计算机。它目前还是虚拟的。如果有一天我们把这个二阶无穷虚拟计算机封装到一个计算机中，那么这个计算机便是一个二阶无穷计算机。如果在全球网络中的计算机已经全部是二阶无穷计算机，那么可以把这个网络看成虚拟的三阶无穷计算机。如果把三阶无穷虚拟计算机封装进一个计算机，那么我们就得到了一个三阶无穷计算机。如法炮制，直到更高阶无穷计算机。

研究高阶无穷计算机应当注意如下的基本问题：

- ◆ 研究高阶无穷计算机，首先要研究“无穷”的数学，例如 Martin_Löf 的文献《Mathematics of Infinity》，研究无穷是如何成为一个对象和被构造的。
- ◆ 无穷计算机有无限的空间、无限的时间、无限的处理能力、无限的过程、无限逼近以及无限的计算速度，同时还可以获得无限的不确定性。必须借鉴相对论思考问题的方法，来研究网络世界中的问题，去建立网络世界中

统一的时空概念体系和环境。要把时间、空间、过程、速度、逼近和不确定性等都理解为相对的，可以相互转换。在网络的时空概念中存在着奇异特征。计算模型必须有这样的时空观；而经典的可计算理论是静止和绝对的，没有这样的时空观。

- ◆ 上述的“无限”的特性是具有“阶数”的，即一阶无穷大 (∞^1)，二阶无穷大 (∞^2)，三阶无穷大 (∞^3)，乃至无穷阶无穷大 (∞^∞)。同样，也有一阶无穷小 (∞^{-1})，二阶无穷小 (∞^{-2})，三阶无穷小 (∞^{-3})，乃至无穷阶无穷小 ($\infty^{-\infty}$)。
- ◆ 在上述的高阶无穷计算机模型中，再去研究那些在经典计算模型计算机上已经有结论的一切问题和从来没有研究过的问题，会给我们展开一个无限的研究天地。
- ◆ 把这种高阶无穷计算机与在本书后面谈到的群体计算理论联系起来，相信读者会有更深刻的理解与认识。

高级无穷计算机将逐步逼近人类的智慧结构：实数空间。

第 4 章

程序设计语言

4.1 300 多种程序设计语言

在计算机机器语言和汇编语言时代，编写程序的手段非常原始，采用纸带和卡片，调试程序非常困难，生产力低是显然的。这个时期，生产程序的模式是非交互形式的，需要人机交互的工作方式是当时的主要问题。到了 20 世纪 70 年代初，有了计算机终端，人机交互改变了编写程序和调试程序的手段。但是，这个时期的程序设计是自由风格的，对程序设计和程序设计语言的理解还处在低级的状态上，每人每年编写 3000~5000 行正确源程序，软件生产力仍然非常低下。这种状态甚至持续到了 20 世纪 80 年代初。在那个时期，业界普遍认为程序错误是阻碍程序生产力提高的主要原因。

如何才能少犯错误？当时的计算机科学家认为问题主要出在程序设计语言上（笔者也这样认为），在这种思想指导下，当时的计算机科学家居然搞出了 300 多种语言。世界上第一个高级程序设计语言是 FORTRAN。随后，又出现了 ALGOL60（1960）、ALGOL68（1968）、PL/1 语言、COBOL 语言、PASCAL 语言、并发 PASCAL 语言、BASIC 语言和 C 语言等。下面介绍一下其中的几种程序设计语言。

FORTRAN 是面向计算与应用的程序设计语言、模块式的程序结构，有主程序与子程序之分，可以编写大型的应用软件，但不能用于系统软件开发（例如操作系统的程序设计）。当时的系统程序设计主要依靠汇编语言，待到 C 语言问世后，又主要依靠 C 语言进行。尽管 PASCAL 语言和并发 PASCAL 语言具有许多优秀的特性（例如类型、递归、规范语句结构等，并发 PASCAL 语言甚至已经有了类、进程、监视器等先进语言结构），受到了

一些从事算法的编程人员和计算机科学学者的喜欢，但是，由于其结构的嵌套性，它们不能适应面向事务处理的应用软件（尤其是大型应用软件和系统软件）的开发，因而被市场淘汰了。

1980 年，在美国国防部的支持和参与下，推动了 ADA 程序设计语言及其巨大的软件工程 CASE 计划的制定。对于一个程序设计语言来说，ADA 语言从规模上来说是空前的，几乎全世界的计算机科学家都参与其中，企图成为最后的程序设计语言。ADA 程序设计语言几乎集中了已有程序设计语言的所有优势，例如类型、子类型、派生类型、抽象数据类型、类属、程序包、强类型机制、类型转换、合理完善的引用实现机制、安全机制、进程、进程类型、进程同步、进程通信和不确定性等特性。ADA 程序设计语言不仅支持说明的开发，同时也支持实现的开发。至今看来，ADA 语言说明方式也是十分优秀的，甚至有时看到 UML 语言形式，笔者第一个想到的就是一个可视化的 ADA 说明语言，有时还觉得 UML 没有 ADA 说明语言杰出。ADA 的设计者不仅考虑了软件重用技术，还考虑了软件自动生成（当然，最后没有实现）。ADA 语言计划和 CASE 计划推动了一大批工具语言和软件开发工具的问世。在 20 世纪 70 年代和 80 年代初，笔者曾经是我国 ADA 语言的立项者和推动者。但是，随着 ADA 语言计划的落实和实践的检验，笔者对 ADA 语言的计划，尤其对 ADA 语言的实现开发计划转而持批评态度，反而开始喜欢 C 语言了。

在计算机科学中，应当引起关注的程序设计语言是 LISP 语言和 PROLOG 语言。LISP 语言是函数程序设计语言，为这种语言设定了 LISP 计算机。因为没有操作系统的支持，或者说并发程序设计函数缺少资源管理等操作系统功能，它失败并退出了历史舞台。PROLOG 语言是逻辑程序设计语言，曾经成为人工智能的程序设计语言，风靡一时，被日本人作为第五代计算机的模型

语言，随着第五代计算机计划的失败，逐步退出了历史舞台。

进入 20 世纪 90 年代后，产生了一些新型的语言：JAVA、XML 和 UML 及其变种系列。

4.2 永恒的 C 语言

C 语言诞生于 20 世纪 70 年代，与 UNIX 操作系统相伴而生。现在，C 语言与所有的操作系统都有联系。C 语言是公共、开放的程序设计语言，所有的大学生都学习它。C 语言是面向系统程序设计的。具有变量、函数指针以及合理的数据类型定义，使 C 语言具有广谱性，可以说它几乎无所不能，可以编写系统软件，也可以编写应用软件。在现代程序设计实践中，无论使用什么软件开发工具，C 语言都是不可缺少的。C 语言自身的发展适应时代的要求，在 20 世纪 80 年代初，扩充了类（class）的概念，并产生了 C 语言的发展版本——C++。C 语言与操作系统的系统服务结合，又提供了更多的并发程序设计能力，甚至设备驱动程序也可以完全用 C 语言开发。在可视化技术发展的冲击下，20 世纪 90 年代实现了可视化的 C 语言与可视化的 C++ 语言，例如 VC++。C 语言的上述进步，大大提高了 C 语言的开发能力。软件重用技术与面向对象技术推动各 IT 厂商提供了大量 C 语言类库和软件重用库，使 C 语言的开发能力大大提高。由于 C 语言的公共开放性、广谱性及其巨大的开发能力，从市场和技术发展两个方面来看，我们都可以说 C 语言是永恒的，至少将伴随电子计算机一直存在下去。

第 5 章



计算机系统

5.1 单机与多机操作系统

计算机操作系统从计算机设备的管理程序开始，通过单道管理程序、多道管理程序发展为现代的操作系统体系。早期的操作系统主要作为计算机系统的资源管理环境和运行支持环境。在操作系统应用方面，走过了字符终端、图形终端和网络终端的管理阶段，发展到操作系统完全建立在可视化应用与管理基础上的当前阶段。现代计算机操作系统作为 IT 系统的五大环境：软件的运行支持环境、系统的管理支持环境、系统的应用支持环境、系统的开发支持环境和系统的集成支持环境。计算机操作系统既然作为上述五个方面的支持环境，理所当然地作为系统的平台，是 IT 的核心。但是，在 IT 技术快速发展的浪潮中，操作系统发展得相对缓慢，它的每一次发展都要承载“兼容性”的负担。

现代单机操作系统的核心概念是进程和子进程或进程和线程，这种进程、子进程或线程在操作系统中有其相应的类型概念或结构框架。有了这种框架，用户就可以很方便地建立进程、子进程或线程这些实体。计算机运行支撑也是以进程、子进程或线程为单位在进行的。

现代计算机操作系统，例如 UNIX 和 Windows，实际上都与美国原 DEC 公司的计算机 PDP 和 VAX 系列的操作系统相关。VAX/VMS 操作系统（1983）是 DEC 公司的拳头产品之一，它是军用操作系统，可靠性极高，一旦配置好，几乎不会发生停机事件。它既是分时系统，又是实时系统，系统服务全面，称得上是世界上最好的操作系统。但是，DEC 公司长期将它捆绑在 VAX 计算机上不开放，结果造成了自己的失败。在笔者看来，DEC 公司最有价值的是 VMS，而不是 DEC 的硬件服务器和小型计算机。UNIX 出于 VMS 操作系统的前身，即由美国 AT&T 公司的技术

人员在 DEC 公司的 PDP11 计算机上开发的操作系统，于 1973 年正式推出，起初主要用于大学中，主要是分时系统。早期 UNIX 的原始商业版本主要有四个，第一个是 AT&T 的 SYSTEM V 版本，第二个是伯克利大学（Berkeley）的 BSD 版本，第三个是由 DEC 公司、IBM 公司和 HP 公司等组成的“开放软件基金会”的 OSF/1 版本，第四个是以 Carnegie Mellon 大学的 MACH 操作系统内核为基础的 UNIX 系统。现在，UNIX 操作系统变种为 Linux 系统，由于其开放源代码，得到了越来越多人的重视。

操作系统是信息系统建设的基础，信息化的发展需要操作系统的标准化和规范化。各主要计算机厂商的操作系统的明显差别严重阻碍了 IT 技术和产业的发展，是 IT 系统“烟囱林立”的根本原因，是阻碍系统资源和信息重用、共享、交换、互联、互通与互操作的本质原因，也是 IT 产品成本居高不下的主要原因。业界有识之士也曾为 UNIX 的统一付出巨大的努力。国际标准化组织提出了 POSIX 操作系统标准，这个标准分级支持分时操作系统，也支持实时操作系统。但是，由于 UNIX 操作系统的派别不同，也由于 UNIX 操作系统是基于 C 语言源代码程序的，系统之间的连接是静态特性的，为了实现 UNIX 操作系统动态连接和系统集成，提高 UNIX 操作系统的其他服务特性，不同 UNIX 厂商提出了不同的解决方案，导致计算机厂商的 UNIX 操作系统依然千差万别。正因为如此，OMG 组织的模式驱动体系结构（MDA）研究居然首先实现了非平台（操作系统）依赖的系统开发，然后实现了依赖平台转换。实际上，统一操作系统是非常困难的。这使得 MDA 的计划不仅仅解决软件自动生成，还解决软件在多平台之间的可移植性。请注意，这种平台并不是解决互操作性的信息化平台。

现代操作系统在用户界面上采用了视窗技术，尤其终端操作

系统和操作系统管理系统都使用了 GUI 截面和视窗技术。需要提醒读者注意的是，要区分不同操作系统应用领域：

- ◆ 要区分实时与分时操作系统。
- ◆ 要区分服务器和终端、客户器或个人计算机操作系统。
- ◆ 要区分计算与事务处理计算机的操作系统。
- ◆ 要区分应用服务器、大型数据处理服务器和网络服务器的操作系统。
- ◆ 要区分通用操作系统和专用嵌入式操作系统。

之所以要做上述区分，是由于这些不同应用领域对操作系统技术体制、功能、性能和配置的要求有非常明显的区别。相当数量的信息系统建设的功能与性能发挥不好，就是由于在操作系统的选型上没有认真考虑造成的。

5.2 应当开发多代理网格操作系统

在传统的单机或多机操作系统上建立代理或多代理系统是一件非常复杂的事情。由于代理是由一组进程、子进程或线程依据代理协同机制和功能机制建立的规范组合体，在这样的操作系统环境下，如果需要创建一个代理，用户可根据代理的结构，在应用层面使用创立进程、子进程或线程的方法，通过专门的服务将它们捆绑在一起，通过同步和进程通信机制实现代理内部结构或者代理之间的信息交换。同样，如果构建一个多代理体系，也必须通过许多微观的服务和动作，才能将许多代理构成一个群体的组织结构。因此，单一计算机上的操作系统通过比较微观的进程或线程概念来建立代理和多代理的群体软件的应用模式，这将是一项非常复杂的工作。打个比方，当代单机操作系统或者说

Windows 操作系统对于开发代理或多代理系统，就如同上世纪 70 年代到 80 年代汇编语言对于程序设计那么困雅。

也就是说，在现代单机操作系统中，没有一个代理结构的类型为用户直接构成代理、多代理和更高级代理群体对象提供服务，也没有支持代理和多代理运行的环境和服务。所以，信息化的发展需要建立一种以代理和多代理为核心概念的网格操作系统，为用户提供方便的代理和多代理建立、运行和控制的便捷服务机制。真正的网格操作系统要系统化地支持代理网格、代理网格平台、多代理、代理网络、规范代理和进程等多层次的系统体系结构形成。我们可以设想，一个使用网络操作系统的用户，可以面向模式和面向对象方便地建立任何需要的群体软件的组织模式体系结构。

我们可以得出结论：计算网格操作系统将代理或多代理概念作为网络系统的核心概念。为了区别旧式的操作系统概念，我们称之为多代理网格操作系统（见图 5-1）。

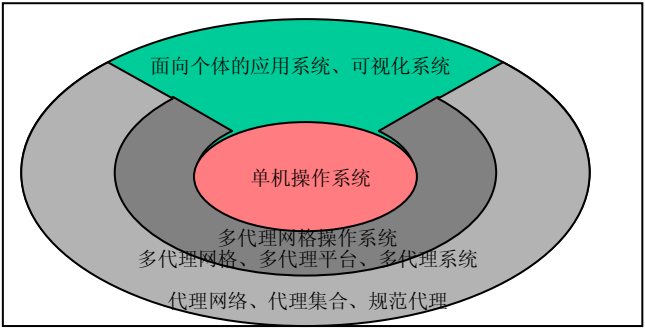


图 5-1 需要多代理网格操作系统

开发多代理网格操作系统是建立第二个微软公司的一次良好机遇。

第 6 章



计算机应用

计算机应用在早期是单机性质的应用，主要是完成计算任务。但是，到网络时代后，计算机的应用主要表现在信息系统应用上。信息系统应用主要划分为管理信息系统（MIS）和管理系

统 (MS)。管理信息系统是主要管理人类世界的人类活动以及人类资源的信息系统，而管理系统（包括监控系统）主要管理网络世界中的设备、硬件、软件、系统、网络、主体、客体和数据等资源。

6.1 可视化与代理化的管理信息系统 (Agent MIS)

管理信息系统的发展大致经过了如下 5 个阶段，分别表示管理信息系统在当时遇到的主要问题：

(1) 主机应用模式的管理信息系统发展阶段。在 20 世纪 70 年代开始的管理信息系统主要是采用面向主机系统，采用字符终端的信息系统，主要用于业务计算和一些项目的管理。

(2) 二层 C-S 应用模式的管理信息系统发展阶段。二层 C-S（客户器-服务器）应用模式采用了计算机局域网分布式概念，起源于 20 世纪 80 年代，采用图形界面和菜单方式实现操作，开始启动全面可视化进程，计算机可用性得到了很大提高。

(3) 三层 C-S 应用模式的管理信息系统发展阶段。这一阶段希望将表示服务、应用服务和数据管理服务全面分离，起源于 20 世纪 90 年代。这种分离最大的好处是表示服务实现了规范化，将多个管理信息系统的数据管理综合在一起，建立数据仓库体系，实现数据共享。

(4) 互操作性平台式综合管理信息系统发展阶段。面向多厂商开发应用系统，实现系统之间的互联、互通与互操作。在 20 世纪 90 年代后期，纷纷开始建立互操作性平台系统，以解决企

业或更大范围内的综合业务管理问题。建立以浏览器或门户系统（Portal）为工作界面的、多种模式的互操作性平台或者数据交换平台系统，连接与集成更大的系统。

（5）主动模式代理管理信息系统（Agent-MIS）。前面谈到的管理都是人的直接管理或人通过管理信息系统提供的信息对人类行为的管理，但在面对大范围管理任务时，这种管理模式会越来越没有效果。因此，需要建设更大范围的互操作性平台，在信息采集、信息处理、网络服务方面采用代理技术，提供代理服务和主动服务的新模式。

管理信息系统主要是管理人类行为与活动的信息系统，例如财务管理、资源管理、制度管理、服务管理、产品管理、渠道管理、培训管理、维护管理和用户管理等。当代管理信息系统的基本特点是：终端人机界面是可视化的，数据主要通过人工录入、卡扫描录入和票据录入等方法录入。管理信息系统也开始采用过去仅仅应用在管理系统、监控系统等领域的代理化服务方法，例如，对管理信息系统的设备、网络、信息系统的安全管理与防护，对运营中行为与内容的监管，无人值守和虚拟管理员、操作员等都广泛地采用代理化服务技术，而且对智能化要求越来越高。

6.2 管理、监控系统和自动化控制系统

在管理信息系统发展的同时，管理系统（监控系统）也得到了阶段性的发展。这些发展阶段大致划分如下：

（1）单点对象的管理系统发展阶段。被管理对象是单点、个体的，没有组网。监管主体是固定结构的、非组网的，所以管理对象与主体之间不可能离得太远。

(2) 局域可视化管理系统直接面向对象实施管理的发展阶段。通过可视化管理系统对设备、系统等直接实施管理。例如，我们经常碰到的网络管理、计算机系统管理、数据库管理、资源管理及对各种物品、资源、财产或人的物理行为的管理，都属于这类管理范畴。需要一些电子化设备的辅助，管理工作本身是自动或半自动的。此时的信息系统不仅提供管理信息，还承担全部或部分管理任务。

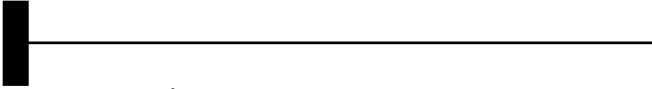
(3) 动态与代理化的管理系统直接面向对象实施管理的发展阶段。通过代理系统实施管理，在管理信息系统、可视化管理系统、业务系统、生产系统、办公系统上安装管理代理系统，对被管理的对象（包括设备、系统、资料、信息、人的网络行为和软件的网络行为）实施管理，具有充分的管理能力，而且可以做到管理体系与运营体系分属不同机构，从而确保管理的合法性与有效性。例如，对计算机网络和电信网络的管理就使用代理技术，其管理协议为 SNMP 和 CMIP。由于这种代理技术不断扩展的自治性和移动性，其主动特性、可管理性和可监控性也更加突出。

管理系统、监控监测系统和自动化控制与操作系统主要采用代理化方法进行管理、服务、操作和测试，其高层管理依然采用可视化控制技术。数据与信息输入主要通过自动采集器、探针、监视器和传感器等来实现。代理化越来越要求有高的代理性和自治性。管理、测试、控制的范围也越来越大，要组成大范围的网络，建立管理、测试、控制等自动化环节的数据库系统，也可以为分析、查询等提供可视化服务。

6.3 可视化与代理化服务技术的融合

可视化服务技术与代理化服务技术是两大类基本技术体制，这两种技术体制在当代信息化建设中越来越多地被联合使用和相互融合。这两大技术范畴相互融合必然带来了当代信息化系统体系结构两个最基本的层面：**MIS** 层面和 **MS** 层面。可视化服务技术范畴与代理化服务技术范畴反映了人类世界中的行为或活动与网络世界中的行为或活动的关联性关系，也反映了在信息系统提供的服务中，有些服务是需要人类自己操作的，有些服务是可以和应当由代理来完成的。

第 7 章



网络系统与网络世界

7.1 走向多网融合的网络世界

与信息化相关的网络系统包括通信网络、计算机网络和广播电视网络。如下的一些观点和分析，基本上也是 QNS 工作室孙玉院士的意见（如有理解偏差，由笔者负责）。

电信业有百年的历史，全球电信网络已经有 10 多亿的用户，除传统的电话业务外，还开辟了数据业务和多媒体信息等业务。进入 20 世纪 80 年代后，提出了综合业务数字网（ISDN）及宽带综合业务数字网（B-ISDN），企图把计算机和电视业务拉入到电信网络中。电信网络具有有线、无线、移动和卫星多种传输体制，以及分组、信元（ATM）和帧中继等多种交换体制。从其传输技术体制看，电信业的发展走过了一条从“确定复用”与“有连接寻址”到“统计复用”与“有连接寻址”的业务发展道路。

计算机网络在早期划分为广域网（WAN）和局域网（LAN）。有意思的是，计算机广域网的开发在先（20 世纪 70 年代），而计算机局域网的开发在后（20 世纪 80 年代）。在 20 世纪 80 年代，谈起计算机网络，主要是指 IBM 的 SNA 和 DEC 的 DNA（DECnet）网络，后来国际标准化组织提出了 ISO/OSI 开放网络系统 7 层协议，开始规范网络的发展。在 20 世纪 90 年代，把 20 世纪 70 年代开发的 ARPANET 网络改名为互联网，网络技术开始了飞速的发展。互联网采用 TCP/IP 网络协议，互联网协议到了 IPv4 才得到应用。IP 网络往下可以依赖计算机局域网（包括以太网和 ATM 网络等，在局域网发展中出现过许多过渡技术，例如 FDDI 和令牌网络）和计算机广域数据网络。在网络市场的自然牵引以及率先参与互联网市场的企业的炒作下，ISP 和 ICP 纷纷出笼，互联网的浏览器和 WWW 技术相继问世，促进了互联

网的飞速发展。许多学者为了在互联网上改变 IPv4 网络上的地址空间紧张的局面和实现多媒体信息的传输和处理，做了大量的工作。经过多年的努力，IPv6 协议于 1995 年应运而生，适应了网络更大的协同范围。但是，这并没有刺激 ISP 对 IPv6 感兴趣。电信运营商认为，IPv4 网络通过 DNS，CIDR，NAT 和 VOIP 等措施，实现了 IP 电话和视频业务。可以说，IPv6 提出的问题基本得到了解决。美国有人呼吁启动宽带 IP 网络计划 NGI（美国下一代互联网计划），它采用 DWDM 密集波分复用技术。在我国，也建设了示范高速 IP 宽带网。近年来，互联网的 IPv6 还没有得到应用，在基本的交换与路由的规则方面却发生了根本性的变化。在电信领域中利用标签技术改进交换、路由和寻址技术。在 20 世纪 90 年代后期到本世纪初，IETF Network Working Group 提出的多协议标签交换（Multiprotocol Label Switching，MPLS）提高了网络传输的效率和质量，甚至提高了网络传输的安全机理。IETF's IPng Working Group 也在积极从事类似的工作，为 IPv6 提供数字标签。另外，主动网络的研究者也积极为其提供活性标签技术研究成果。计算机网络与互联网的基本传输机制走过了一条从“统计复用”和“无连接寻址”到“统计复用”和“有连接寻址”的业务发展道路。

广播电视网络在我国有 20 多年的发展历史，已成为世界上第一大有线电视网络，基本实现了全国城市范围的电视网络。该网络采用 SDH/ATM 体制，主干网络为 2×2.5 Gb/s，全长数十万公里。电视界依靠其网络带宽、网络普及率高和掌握着重要信息源的优势，也提出了类似电信界 ISDN 的全业务网络（FSN），来实现广播电视的数字化，企图将电信、计算机网络的业务纳入到电视网络业务中，企图抢夺 IP 市场，争夺电话业务、文件、电子邮件和视频点播业务（VOD）。广播电视网络虽然目前还处在“确

定复用”与“无连接寻址”的状态中，但电信与计算机网络的发展道路为其发展提供了借鉴与依据。

通过传统的电信网络 ATM 技术与互联网 MPLS 技术，逐步使不同网络技术体制结合起来，走多网融合的道路是必然的趋势。电信、电视、计算机领域的专家们认识到，从技术体制和经济两个方面研究，由一个技术体制综合所有业务是不现实的，应当走多网融合的道路，发挥不同网络技术体制的服务质量和传输效率的优势。多网融合之所以成为可能，是由于有如下的技术基础：

- ◆ 数字化技术的发展和全面采用，使电话、数据、图像信号均采用统一的编码传输和交换。
- ◆ 光通信技术发展为综合传送各种业务信息提供了必要的带宽和传输质量。四通八达、无所不及的光纤网络是多网融合的传输平台。
- ◆ 软件技术的发展，使得多网融合在终端业务上趋向一致。系统的软件含量已经超过硬件。
- ◆ 统一的 TCP/IP 协议被各种网络采用。
- ◆ 多网融合的实际内涵可以归纳为：技术融合、业务融合、市场融合、行业融合、终端融合和网络融合。

近年来，国际 ITU 提出了全球基础设施（GII）的发展思路，就是把电信、电视和计算机网络相融合的发展计划。

7.2 可信信息基础设施网络的基本要求

网络安全是信息基础设施新提出的尖锐问题。过去，对于网络，主要考查网络传输质量与效率，但是必须把网络安全纳入到新的关注范围内。最值得注意的是建立可信网络体系，实现可信

结构、可信信道、可信传输、可信交换与路由、可信定位与追踪、可信接入、可信应用、可信管理、可信认证与监管。

中国信息安全产业规划的可信网络世界体系结构框架（TCAF）对基础信息网络定义了可信网络平台（TNP），例如可以由如下的系统所组成：

- ◆ 可信独立的网络管理、监管和控制网络信道体系，实现网络管理与用户分线传输的结构。
- ◆ 信道安全检测与反插播系统。
- ◆ 支持标签交换、路由和信息传输踪迹确认系统。
- ◆ 可信与安全的网络服务协议系统。
- ◆ 网络定位与追踪系统。
- ◆ 可信网络接入系统。
- ◆ 可信网络应用（安全标签分类业务应用）平台与系统。
- ◆ 可信网络管理平台与系统。
- ◆ 网络行为与内容的监管、认证平台与系统。

在基础网络中，采用数字标签技术是一个正确的选择。首先，在通信传输中使用数字标签技术，这已经表现在采用标签路由与交换（MPLS，APL）上。其次，可以采用网络设备或系统强制的标记上“印鉴标签”技术，使被传输的信元在经过的路径中被盖上设备标签印鉴。如果还希望在传输信道上进一步加强安全措施，可以采用活性客体标签（代理做标签）和活性密钥等活性客体（邮差、保镖）的技术以及信道伴侣与系统伴侣（警察）技术。

在基础网络上应用可以采用信息安全标签技术（例如美国国防部的 RFC 1108(RIPSO)、CIPSO、互联网网络工作组 RFC1457、ISL 和美国商务部/国家标准 FIPS 188 等），还可以采用信息交换标签技术（XML 和 DAML 等）。

第 8 章



软件工程与面向对象的程序设计

软件工程诞生于软件危机时期。软件危机的原因主要是正确程序的生产力处于较低的水平上。当时，人们认为问题主要在于程序设计语言，这个时期曾经有人说程序设计是艺术（Knuth），后来也有人说程序设计是科学，最后说程序设计是工程。在这个时期，研究程序设计方法学是焦点。在众多的方法学研究中，结构化程序设计（模块化、层次化、无 GOTO、自上而下、逐步求精，其实在我们的程序设计实践中，还得加上一条：不要乱停机（写正常或异常的停机或出口语句））和面向对象的程序设计最为人们所关注。

在相当长的一段时间内，软件工程的主要目标是提高软件的生产力。从 20 世纪 70 年代的每人每年编写 3000~5000 行源程序，到现在每人每年编写百万行以上的源程序，提高了三个数量级；这主要是由于随着软件工程的发展，产生了一大批软件开发工具。对提高软件生产力做出重要贡献的措施包括：软件工程方法、面向对象技术、软件重用技术、可视化技术、部件技术和类型程序设计方法，以及基于这些方法的开发工具。软件危机似乎得到了初步解决，但是新的问题又出现了：我们面临着信息平台危机和系统集成问题。人们寄希望于操作系统的统一，但是 UNIX 操作系统统一的困难，几乎使人们失去信心。从网络建设一开始，人们就希望实现信息共享、交换、协同和控制，但是在异构网络上实现互联、互通和互操作非常困难。后来，按照市场的自然规律，互联网问世并发展起来，这时，人们以为网络可以统一了，互联、互通和互操作该实现了。但是，问题还依然存在，人们又提出了公共操作环境（COE）计划和多网融合计划。在实现这些计划时，信息安全的问题日益突出，亟待我们去解决。在解决这些问题期间，出现了多次过热的现象：语言热、人工智能热、新一代计算机热和目前的互联网热，这些热从技术发展的角度来看，也许有好处；但从应用的角度来看，只是一些人的尝试。

一些程序设计方法学代表着软件工程理论与实践的进步，下面分节进行介绍。

8.1 面向对象的程序设计 (OOP)

面向对象 (OO) 的程序设计技术和部件软件系统技术是当代主要的软件技术成果。OO 技术根据软件的开发阶段划分为面向对象的分析 (OOA) 技术、面向对象的设计 (OOD) 技术与面向对象的程序设计 (OOP) 技术。OOP 技术已经有近 40 年的历史了，最早源于 Simula-67 模拟语言的 class (类) 技术。20 世纪 80 年代初，Ada 语言的定义引入了抽象数据类型 (ADT)、派生类型、子类型及软件包 (Package) 机制。同样，在 80 年代初，为 C 语言引入了类概念，当时称之为具有类的 C 语言，后来演变成了 C++ 语言。面向对象的程序设计达到的主要技术成果是软件的重用性，主要是一种自下而上的软件组成方法，但与企图达到的软件自动生成目标还存在距离。

虽然面向对象的技术得到了大多数跨国公司的关注，但是取得显著成果的是微软公司和 OMG 组织。在软件重用方面，通过建立源程序重用库、目标程序重用库、运行时间库和部件库，不仅实现了源程序重用，还实现了二进制代码的重用。更重要的是，OLE 还是一个成功的引用实现，为多厂商、多技术体制的产品与技术提供了具有互操作性的平台。在面向对象的技术方面，许多软件公司实际上将它们作为标准使用。

8.2 面向模式的程序设计 (MOP)

软件工程领域也在研究软件体系结构,OMG 机构提出了模式驱动的体系结构 (Model Driven Architecture, MDA), 目的在于解决跨平台应用的兼容性和软件代码的自动生成问题。用户可以利用这种方法保护自己在应用开发上的投资, 整合资源, 关注历史开发应用, 平滑处理各种 COTS 应用。在这个体系结构研究中, 需要使用 UML (Unified Modeling Language, 统一建模语言) 和 XML (eXtensible Markup Language, 可扩展标记语言)。UML 是继 20 世纪 90 年代开展的如火如荼的 OOA, OOD 与 OOP 之后的一种建模语言, 其研究开始于 1996 年, 到 20 世纪末完成, 产生了大量的标准化书籍和商用工具, 开展了广泛适用训练。建立一个 MDA 应用, 开始于平台独立模式 (Platform-Independent Model, PIM), 不考虑技术细节, 表示业务功能和行为。一个细致的模式, 用 OCL 表示前置条件和后置条件, 用行为语言描述语义, 其建立步骤可以描述如下:

1. 产生平台指向模式 (PSM, Platform-Specific Model)。MDA 工具运用一个标准实现从 PIM 到 PSM 的映射。代码部分自动生成, 部分手工编写。通过 OMG 标准实现 PIM 到指定的中间件技术的映射。
2. 映射到多种开发技术。
3. 产生实现。MDA 工具为开发者产生所有或绝大多数实现代码, 将 PSM 映射到应用接口、代码、GUI 描述符和 SQL 查询等。
4. 集成应用沉淀和 COTS。MDA 工具为在新平台上再集成提供逆向工程自动模式发现。逆向工程将已有应用映射到一个模式, 并重新实施调试。

5. 自动生成桥。桥生成被公共应用模式简化, 简化了跨企业混合应用的创立。

OMG 的 MDA 关注的是软件面向多厂商平台以及软件可移植性和兼容性这样一些概念, 通过平台独立向平台指向转换, 使一个应用软件可以在不同平台上运行。显然, 这些关注点是面向应用软件的, 是传统软件工程的关注点, 站在开发者的立场上。现在看来, 这种从功能与业务需求出发建立一种开发软件体系结构生产系统是 OMG 的 MDA 追求的目标。显然, 在当代信息化中, 互操作性与安全性成为了主要的关注点, 对软件可移植性和兼容性的关注程度已经不那么高了。

8.3 面向类型的程序设计 (TOP)

面向类型的程序设计 (Type Oriented Programming, TOP) 方法学, 是笔者在德国数学家 Martin Löff 提出的著名的类型理论 (Type Theory) 和抽象数据类型 (ADT) 理论基础上, 进行了适合程序设计与软件开发的修改而提出的。作为 863 项目, 在 20 世纪, 有一支素质很高的软件和系统人员团队进行类型程序设计方法学这项工作。这可以从笔者的《形式语义学基础与形式说明》和《实用类型程序设计》等著作中看出, 笔者与几十位学生与弟子在 VAX 计算机系统中共同完成了这项科研与开发工作。这项工作在 20 世纪 80 年代中后期到 90 年代初完成, 并取得了成功。采用类型程序设计理论与方法实现的软件自动生成, 生成的代码质量好, 程序效率高。一个熟悉该理论与方法的高水平的程序设计人员, 在类型表达式编译系统的支持下, 将一个软件工程师的软件生产力提高到“百万行源代码”的水平是可行的。类型程序设计方法学强调将自上而

下（面向模式）与自下而上（面向对象与部件）两种方法结合起来。

类型程序设计方法学认为：

- ◆ 世界是类型的，类型是构造的，类型是层次的，低层次是高层次的成员。
- ◆ 应用的开发是类型的开发，软件的设计是类型的设计。
- ◆ 程序设计语言预定义类型太少，要把类型概念进行到底。
- ◆ 产业程序设计的指导思想是，选择有数学支持的概念。例如，我们可以在 `template`，`package`，`class` 和 `type` 等概念中选择。类型程序设计选择的是 `type`，因为一个 `type` 是一个代数（类别代数）；也就是说，`type` 概念对语义进行了规定，而且这种语义规定是通过大学的公共教育得到的。而 `template`，`package` 和 `class` 这些概念失去了数学的支持（没有数学家研究），还要研究自己定义的各种说不清楚的问题，而且存在很大的任意性。对于企业来说，任意性就是高成本。

类型程序设计方法学是支持软件重用、软件自动生成和智能化应用的重要基础，同时也是面向并发程序设计的方法学和面向代理（主体）的程序设计方法学。类型程序设计方法学在软件重用、软件自动生成和软件智能化方面都有自己的特点。

1. 便捷重用

“穿上规范的类型外衣”的类型程序设计使重用资源规范化、分类化和简单化，是低成本软件重用技术。支持软件重用的技术规则主要有：

- ◆ 选择类型作为设计的基本单位与概念，应用与设计是类型的应用与设计。

- ◆ 追求最好的可读性、可靠性、可维护性、可培训性、非个人特色的程序设计原则。
- ◆ 追求最低成本的设计原则。
- ◆ 建立工程师可读的类型库（ADT，IDT 和 dll 库等）。
- ◆ 类型数据结构。必须用 `typedef` 定义类型变量指针。
- ◆ 类型常数。例如，必须利用 `#define` 为常数命名，在程序设计中没有常数出现。
- ◆ 类型变量。变量命名用类型名与变量名连接的形式，例如 `PFN$ALLOCATE`。变量与函数要标记为 `public`，`protected`，`private` 或 `package`，不用全局变量和静态变量。
- ◆ 类型函数。函数命名用类型名与函数名连接的形式，例如 `stack_push`（或 `stack$push`）。函数必须有返回值，必须无副作用。
- ◆ 参数规则。规范参数变量和指针、参数调用模式（`in` 模式），参数传递机制（`call_by_value` 和 `call_by_reference`）和形参与实参之间的调用关系（按位置还是按名称）。
- ◆ 系统服务的引用规则。引用系统服务必须返回系统服务是否给用户提供返回状态。
- ◆ 表达式规则。例如，使用显式类型转换，尤其是在赋值语句的右边表达式前使用。
- ◆ 语句规则。例如，不要乱用 `goto`，不要忘了在循环语句中使用 `break` 等语句，在 `switch` 语句中出现异常情况时使用 `default` 语句，尤其要控制 `stop` 语句与 `exit` 语句，停机语句要放到异常处理中。
- ◆ 注释规则。包括类型注释、函数注释、语句注释和表达式注释。

2. 软件自动生成

软件自动生成理论、方法与技术实际上是说，给定一个说明语言，根据这个语言，通过逐步的类型分解实现软件的自动生成。这种自上而下的划分与推导工作的基础是什么呢？是自下而上的类型组合与构造。自上而下的推导和自下而上的类型组合与构造的结合点是软件自动生成的关键。也就是说，进行楼房设计时，从需求开始不断地分解、推导，不应该推导到物质组成的最小单位；我们认为，当推导到钢筋材料、水泥、砖木材料、管道材料、电器材料和装饰材料等分类产品时便完成了设计工作，或者认为设计落在了实处。同时，还有一种供应机构为上述材料的分类产品进行自下而上的设计，用比较简单、原始的物质设计、生产这些材料产品。软件自动生成也是同样的道理，我们必须有一个自上而下的划分推导的过程。从直觉上看，采用逻辑语言似乎是描述这个过程的最好方法。但是，在数理逻辑中，语言是从研究数学本身体系的完整性出发的，过于基本，作为自上而下的设计还过分简单。总之，结论是不适用，因为这种数学语言本不是为程序设计服务而产生的。推导软件，不能仅仅有推导的规则和设计原则，还必须有“建筑材料”，即推导的分类基础：类型。于是，我们明确了系统划分是类型的划分，系统分解是在类型中的分解，系统推导是在类型中的推导。例如，给定任意一阶谓词演算的合适公式，假定谓词文法表示为：

$$p ::= a \mid p \wedge p \mid p \vee p \mid p \rightarrow p \mid p \leftrightarrow p \mid \neg p \mid (\exists x \in A)p(x) \mid (\forall x \in A)p(x) \mid (p)$$

其中， a 是原子公式， p 是一个谓词， A 表示一个集合。

能否自动从上述的谓词说明中转换出程序呢？命题演算很简单，我们暂且不去讨论它，而只讨论两个量词约束的谓词，看 $(\exists x \in A)p(x)$ 及 $(\forall x \in A)p(x)$ 如何写出程序。请看如下的程序实现结构。

1) $(\exists x \in A)p(x)$:

```
{for all x∈A
do if p(x)
    then return(true)
    else skip
fi
od;
return(false)
}
```

2) $(\forall x \in A)p(x)$:

```
{for all x∈A
do if p(x)
    then skip
    else return(false)
fi
od;
return(true)
}
```

对于多个量词约束的情况，程序生成的框架可以是：

3) $(\exists x \in A)(\exists y \in B)p(x, y)$:

```
{for all x∈A
do if
    for all y∈B
    do if p(x,y)
        then return(true)
        else skip
    fi
    od;
return(false),
then return(true)
else skip
fi
```

```
od;  
return(false)  
}
```

从生成的角度看，软件的关键在于 `for all $x \in A$ do...od` 的枚举算法的产生。如果 A 是一个类型，那么对不同类型的 `for all $x \in A$` 的实现程序显然是不同的。于是问题就产生了：是否存在着一种枚举算法程序，对于任意 A 都是有意义的呢？因为只有如此，谓词才有可能转换成程序，否则，我们是枚举不完所有类型的 `for all $x \in A$ do...od` 程序的。

类型程序设计软件自动生成的原理是，在软件重用技术的基础上再增加如下的考虑：

- ◆ 系统是由类型构造（自下而上）和类型分解（自上而下）设计的，这种类型构造和类型分解是由类型表达式表示的。
- ◆ 类型也是构造的，类型的构造是由构造类别代数描述的，其中包括：集合是构造的、函数是构造的、证明是构造的、代数是构造的、类型的范畴是构造的、范畴的范畴也是构造的。
- ◆ 一个类型的生成是通过构造性类型说明语言描述的，类型软件是通过编译程序自动生成的。
- ◆ 提供抽象数据类型到实例类型的自动发生服务函数。

软件自动生成主要使用三个工具：类型表达式编译程序、类型说明语言和实例类型发生函数。

1) 类型表达式的语法定义

<code>typeexpression ::= atomtype-----</code>	原子类型
<code> typeexpression×typeexpression---</code>	乘积类型
<code> typeexpression+typeexpression---</code>	和类型
<code> typeexpression→typeexpression--</code>	映射类型
<code> typeexpression•typeexpression---</code>	连接类型


```

| typeexpressionn----- n 长度表类型
| typeexpression*----- 无穷表类型
| (typeexpression)----- 优先结合类型
|  $\mathbb{P}$ [typeexpression]----- 幂类型
atomtype ::= real | integer | boolean | characters |
string | ...

```

类型表达式的语义概念可以如下定义: 如果 T_1, T_2, T_3 是类型, 那么下面的也是类型:

- ◆ $T_1 \times T_2$ (乘积类型), 其元素为二元组, 即如果 $t_1 \in T_1, t_2 \in T_2$, 那么 $\langle t_1, t_2 \rangle \in T_1 \times T_2$ 。如果 $t \in T_1 \times T_2$, 那么 $\text{first}(t) \in T_1$, $\text{second}(t) \in T_2$ 。
- ◆ $T_1 + T_2$ (和类型), 其元素要么是 T_1 中的元素, 要么是 T_2 中的元素。我们为该类型定义了三个操作: onto, into 和 in。这三个操作的定义如下: 如果 $X = \dots + Y + \dots$, $x \in X$ 并且与 $y \in Y$ 相对应, 那么 $x \text{ onto } Y = y$, $y \text{ into } X = x$; 如果对于 $x \text{ in } Y$, 存在 $x \in X$ 与 $y \in Y$ 相对应, 那么 $x \text{ in } Y = \text{true}$, 否则, $x \text{ in } Y = \text{false}$ 。
- ◆ $T_1 \rightarrow T_2$ (映射类型), 如果 $f: T_1 \rightarrow T_2$, 那么 $\text{dom}(f) \in T_1, \text{cod}(f) \in T_2$; 即如果 $t_1 \in T_1$, 那么 $f(t_1) \in T_2$ 。如果 $t: T_1 \rightarrow T_2, t_1 \in T_1, t_2 \in T_2$, 那么 $t + \{t_1 \mapsto t_2\}$ 表示在 t 中增加一个成分。
- ◆ $T^n = T_1 \times T_2 \times \dots \times T_n$, 其元素是一个 n 元表格 $\langle t_1, t_2, t_3, \dots, t_n \rangle$, 如果 $t \in T_1 \times T_2 \times \dots \times T_n$, 那么 $t \downarrow 1 \in T_1, t \downarrow 2 \in T_2, \dots, t \downarrow n \in T_n$ 。为了表示方便, 有时用 $t \bullet t_1, t \bullet t_1, t \bullet t_2, t \bullet t_3, \dots, t \bullet t_n$ 表示 n 元组的某个对象。
- ◆ $T^* = T^0 + T^1 + T^3 + \dots$, 表示无穷表或不确定长度表。
- ◆ $\mathbb{P}[T]$ 表示幂集类型, 研究不确定性使用的类型表示方法。

2) 类型说明语言的文法格式

可以用于软件自动生成的抽象数据类型(ADT)说明格式如下:

```

type type_name (type_parameters) is
sorts "sorts_structure_definition" end sorts
constructors
    function_name: function_type_expression
    [function_name: function_type_expression]
extensions
    function_name: function_type_expression
    function_name: function_type_expression
    .....
    function_name: function_type_expression
equations for all " object declaration " let
    equation
    equation
    .....
    equation
end type

```

3) 实例类型发生

在上述的类型表达式中做如下的扩充:

```

typeexpression ::= ...
typeexpression ::= ...|type_name(type_value)--实例发生
type_name ::= identifier
type_value ::= type_name|type_name,type_value
type_definition ::= INST type_name (typeexpression)

```

一个抽象数据类型可以表示成:

$$t' = t \cdot (S, \quad , A, O)$$

如果令语义解释函数为:

$$TT: \text{Instdec} \quad ADT \quad TPAR \quad IDT$$

其中，ADT 表示抽象数据类型，TPAR 表示类型参数，IDT 表示实例数据类型，那么实例发生声明（Instdec）的语义为：

$$TT \llbracket \text{type } t' = \text{INST } t(\) \rrbracket (t)(\) = \{ \quad t \ . (S, \ , A, O) \} \\ (t')(\)$$

可以对一个抽象数据类型进行实例发生，例如，对于 stack(TP, DEPTH)，可以通过参数替换得到如下的实例类型：

```
type mystack = INST stack(int,100)
type yourstack = INSTstack(char,100)
```

3 . 软件智能化

我们说过，采用一阶谓词逻辑方法来解决软件自动生成问题有体制上的困难，但是这些逻辑理论方法对软件智能化应用是大有帮助的。笔者对软件智能化的看法是：软件智能化不是通过知识库和规则库等方法实现的，而必须通过软件自身的逻辑能力、归纳能力、证明能力和分析能力等来实现。软件智能化必须将上述能力形式化和可执行化。软件构造和构造数学是软件智能化的原理与基础。软件智能化要求 TOP 的类型是构造的：类型是构造的，类型的构造是由构造类别代数描述的，包括集合是构造的、函数是构造的、证明是构造的、代数是构造的。

软件智能化一般可以在三个方面做工作：

1) 定义逻辑类型

我们可以为软件定义一阶谓词、模态逻辑、时态逻辑、模糊逻辑、多值逻辑的抽象数据类型。

一阶谓词演算的 BNF 定义如下：

P	= A	原子公式
	P P	与操作
	P P	或操作

P P	蕴涵操作
P P	等操作
P	非操作
(P)	括弧
(∀x B)P	全称量词
(∃x B)P	存在量词
(∃!x B)P	惟一存在
(∃(n)x B)P	在 A 中存在 n 个 x 使得 P.....
(A L ∃(n)x B)P	在 A 中至少存在 n 个 x 使得 P.....
(A M ∃(n)x B)P	在 A 中至多存在 n 个 x 使得 P.....
A = true false T relop T	
T = x C T, T f(T)	
relop = < > = /	

其中，量词部分可以做如下解释：

- ◆ $(\forall x \in B)P(x)$ 表示，对于 B 中的所有对象（或成分），P(x) 为真。
- ◆ $(\exists x \in B)P(x)$ 表示，在 B 中存在一个对象（或成分）使 P(x) 为真。
- ◆ $(\exists! x \in B)P(x)$ 表示，在 B 中仅存在着惟一的 x 使 P(x)为真。
- ◆ $(\exists (n)x \in B)P(x)$ 表示，在 B 中精确地存在着 n 个对象（或成分）使 P(x)为真。
- ◆ $(A L \exists (n)x \in B)P(x)$ 表示，在 B 中至少存在着 n 个对象（或成分）使 P(x)为真。
- ◆ $(A M \exists (n)x \in B)P(x)$ 表示，在 B 中至多存在着 n 个对象（或成分）使 P(x)为真。

predicate 类型的一阶谓词逻辑系统可以定义为：

```
type predicate(A) is
sorts predicate,A
```

```

construetors :
    true:  predicate,
    false: predicate,
    A_isformula: predicate,
extensions :
    and: predicate, predicate predicate;
    or: predicate, predicate predicate;
    not: predicate predicate
    impl: predicate, predicate predicate;
    equiv: predicate, predicate predicate;
    forall: A, A, predicate predicate;
    exist: A, A, predicate predicate;
axioms : for all p, p1, p2 predicate and C1, C2(x, i) A let
    1. true = true;
    2. false = false;
    3. p1 and p2 = if p1 then p2 else false fi
    4. p1 or p2 = if p1 then true else p2 fi
    5. not p1 = if p1 then false else true fi
    6. p1 impl p2 = if p1 then p2 else true fi
    7. p1 equiv p2 = (p1 impl p2) and (p2 impl p1)
    8. forall(C1, C2(x, i), p(x))
       = if p(C1)
           then let p(x) = true in
               if p(C2(x, i))
                   then true
                   else false
               fi
           else false
       fi      /* 这个规则消去了全称量词 */
    9. exist(C1, C2(x, i), p(x)) = not forall
       (C1, C2(x, i), not p(x)) /* 最终消去存在量词 */

```

在有限意义下，我们还定义了“惟一存在” ($(\exists! x \in A)P(x)$)、
“存在 n 个” ($(\exists(n)x \in A)P(x)$)、
“至少存在 n 个” ($(A \text{ L } \exists(n)x$

$\in A)P(x))$ 及 “至多存在 n 个” $((A M \exists(n)x \in A)P(x))$ 。在符号证明中，尤其是在无穷意义上，不能给出保证化约终止的替换公式，也就是说，不能给出保证终止的算法程序。因此，在无限意义上，这些公式的语义是“不清楚”的。

在上述基础上，我们可以定义模态逻辑、时序逻辑和模糊逻辑类型。

实例 1：模态逻辑类型 (Modal Logic Type)

在定义模态逻辑类型之前，先简单介绍一下模态逻辑的基本概念。模态逻辑在一阶谓词逻辑的基础上加入了两个模态词： \Box （必然）与 \Diamond （可能），其文法可以定义为：

```
m ::= a
    | m ∧ m
    | m ∨ m
    | m → m
    | m ↔ m
    | ¬ m
    | (m)
    | (∀x ∈ B) m
    | (∃x ∈ B) m
    | ◇ m
    | □ m
    | .....
```

下面，对其语义给出一个形式定义。令 $m: \text{Modal}$ ，定义 U 为一个“可能世界”，令 w 是 U 的对象，并且 U 中的对象具有偏序关系（用 \leq 表示，例如， $w_i \leq w_j$ 表示 w_i 偏序于 w_j ）。

下面，我们给出模态逻辑的语义：

```
SEMANTIC DOMAINS:
w: U = {w1, w2, ..., wi, ...}
TrueValue = {0, 1}
SEMANTIC FUNCTIONS:
```

SEMANTIC RULES:

$$M \llbracket P_1 \wedge P_2 \rrbracket w_i = \text{if } M \llbracket P_1 \rrbracket w_i \wedge M \llbracket P_2 \rrbracket w_i \\ \text{then } 1 \text{ else } 0 \text{ fi}$$
$$M \llbracket P_1 \rightarrow P_2 \rrbracket w_i = \text{if } M \llbracket P_1 \rrbracket w_i \rightarrow M \llbracket P_2 \rrbracket w_i \\ \text{then } 1 \text{ else } 0 \text{ fi}$$
$$M[\Box P_1] w_i = \text{if } (\forall w_j \in U) (w_j \geq w_i \rightarrow M[P] w_j) \text{ then } 1 \text{ else } 0 \text{ fi}$$
$$M[\Diamond P_1]_{w_i} = \text{if } (\exists w_j \in U) (w_j \succcurlyeq w_i \rightarrow M[P]_{w_j}) \text{ then } 1 \text{ else } 0 \text{ fi}$$

根据上述语义定义，我们可以定义如下类型：

constructors:

A\$isformula: \rightarrow modal

necessary: $U, U, U, \text{modal} \rightarrow \text{modal}$

```
possible: U,U,U,modal → modal
axioms:
1~9. (同谓词类型)
10. necessary(y,C0,C1(x,i),m)
    = if forall(C0,C1(r,i),impl(y≥x,m)) then true
      else false
    fi
11. possible(y,C0,C1(x,i),m)
    = if exist(C0,C1(x,i),impl(y≥x,m))
      then true
      else false
    fi
```

实例 2：时序逻辑类型（Temporal Logic Type）

时序逻辑（Temporal Logic）是程序设计语言时序状态的一种特定意义下的模态逻辑，其文法形式为：

```
T ::= a
    | T ∧ T
    | T ∨ T
    | T → T
    | T ↔ T
    | ¬ T
    | (T)
    | (∀x ∈ B) T
    | (∃x ∈ B) T
    | ◇T           always 算子
    | □T           sometime 算子
    | ○T           next 算子
    | T △ T        until 算子
    .....
```

显然，对于时序逻辑的语义，我们只需要解释上述方法的后 n 项，采用实例 1 的语义域及语义解释函数：

$$M \llbracket \Box T \rrbracket w_i = \text{if } (\forall w_j \in U) (w_j \geq w_i \rightarrow M \llbracket T \rrbracket w_j)$$


```

        then 1 else 0 fi
 $M \llbracket \Diamond T \rrbracket w_i = \text{if } (\exists w_j \in U) (w_j \geq w_i \rightarrow M \llbracket T \rrbracket w_j)$ 
        then 1 else 0 fi
 $M \llbracket \bigcirc T \rrbracket w_i = \text{if } M \llbracket T \rrbracket w_{i+1}$ 
        then 1 else 0 fi
 $M \llbracket T_1 \triangle T_2 \rrbracket w_i = \text{if } (\forall w_j \in U) (\forall w_k \in U) ((w_j \geq w_i \rightarrow M \llbracket T_2 \rrbracket w_j)$ 
        and  $(w_i \leq w_k \leq w_j \rightarrow M \llbracket T_1 \rrbracket w_k))$ 
        then 1 else 0 fi

```

根据这个定义，我们可以定义时序逻辑类型：

```

type temporal(A,U) is
sorts temporal,A,U
constructors:
    true:  $\rightarrow$  temporal
    false:  $\rightarrow$  temporal
    A$isformula:  $\rightarrow$  temporal
extenstons:
    and: temporal,temporal  $\rightarrow$  temporal
    or: temporal,temporal  $\rightarrow$  temporal
    not: temporal  $\rightarrow$  temporal
    impl: temporal,temporal  $\rightarrow$  temporal
    equiv: temporal,temporal  $\rightarrow$  temporal
    forall: A,A,temporal  $\rightarrow$  temporal
    exist: A,A,temporal  $\rightarrow$  temporal
    always: U,U,U,temporal  $\rightarrow$  temporal
    sometime: U,U,U,temporal  $\rightarrow$  temporal
    next: U,U,U,temporal  $\rightarrow$  temporal
    until: U,U,U,temporal,temporal,U  $\rightarrow$  temporal
axioms:

```

1~14. (同实例 1)

next 和 until 两个函数的定义公理等式留给读者作为练习。

实例 3：模糊逻辑类型

模糊逻辑是建立在模糊集合论上的逻辑学，其取值范围为[0,1]

实数空间，是非枚举的。模糊逻辑的原子公式是那些值域在 $[0,1]$ 实数空间上的函数，这一点与其他逻辑学不同。

我们可以直接定义模糊逻辑的类型：

```

type fuzzy(A)  is
sorts fuzzy,A
constructors :
    /* A 的构造函数，而 A =  $[0,1]$  的实数空间 */
extenstons :
    and : fuzzy,fuzzy    fuzzy
    or  : fuzzy,fuzzy    fuzzy
    not : fuzzy    fuzzy
axioms :
    1 : a and b = A$min(a,b)
    2 : a or b  = A$max(a,b)
    3 : not(a)  = 1-a

```

2) 项形式符号处理

类型程序设计的函数“施用”构成项表达式，例如，如下的项形式为符号处理提供了方便：

```

push(push(...push(emptystack( ), 1), ...), 5), 6)
emptystack() • push( 1) • push(2) • push(3) • push(4) • push(5)

```

所以，笔者还认为软件智能化必须对计算机科学最基础的概念——“计算”和“替换”进行区别。软件的逻辑能力、归纳能力、证明能力和分析能力等应当是通过符号的处理逻辑能力达到的，需要的数学支持是代数等式理论与项重写理论。代数等式理论与项重写系统是程序证明的重要理论方法，在替换概念的基础上讨论等式的意义，为程序证明提供支持。项重写系统包括：

◆ 顺序项重写系统

◆ 并行项重写系统

为了实现项重写系统，可以定义一个类型化规约语言 (Reductive Language)，并实现它的软件系统。

3) 枚举归纳与结构归纳

如果说在有限集合中，对于 $(\forall X \in A) P(X)$ ，可以采用数值计算的方法来完成：

```
{ for all  x∈A
  do if  P(x)
      then skip
      else return(false)
  fi
od
return(true)
}
```

而对于可枚举无穷集合，采用数值计算方法不能停机。使计算方法具有智能化的方法为归纳方法。

归纳方法：对于可枚举的无穷集合，证明的方法是枚举归纳方法。我们在中学就学过自然归纳法，在集合论中研究序数时，还有超穷归纳方法。归纳原理在数理逻辑中可以作为公理而存在，即：

$$P(0) \quad ((\forall k \text{ Natural}) P(k) \rightarrow P(k+1)) \rightarrow A(n)$$

其中， n 为自然数。

用归纳方法来证明，这时的方法就不是数值计算，而是符号公式的处理。

在计算机科学中还经常谈到另一种归纳方法——结构归纳方法。什么叫结构归纳方法呢？先请看一个例子。

设有一个文法 $X ::= A|F(X)$ ，可以看到，该文法所能接受的语

言为：

$$LX = \{ w \mid w \text{ is accepted by } X \}$$

其中， w 的形式为：

$$F(F(\dots F(A)))$$

显然， LX 也是一个可枚举的无穷集合。对于 LX 中的项，研究其属性与特性时，一般只需从两个方面证明就充分了。设特性 $P(w)$ ，使它对于 LX 中的所有对象都是成立的，那么只要证明：

- ① $P(A)$ 是成立的。
- ② 如果对于 w ， $P(w)$ 成立，那么 $P(F(w))$ 也成立。

就可以说， $P(w)$ 对于 $w \in LX$ 都成立。这样的归纳方法，称为结构归纳方法。

我们将结构归纳方法用于类型中函数的软件自动生成。结构归纳方法还常常被用在证明上，而且这种证明往往在计算机上是可执行的。

8.4 面向视角的程序设计 (Aspect-OP)

我们在第 1 章中曾经讲过，软件需求不仅仅是软件功能提出的，还可以从软件可靠性、软件安全性、软件的可配置和可管理、系统集成提出，也可以为融入更大系统的一致性和互操作性提出。一句话，这种需求或软件的输入/输出条件是从不同关注者的不同关注点提出的。一个软件的代码不仅仅是功能代码，而且包括不同关注点要求的代码。20 世纪 90 年代，美国 IT 产业界首先提出面向多视角的程序设计方法，例如，Kiczales, G.等人在 1997 年提出的面向多视

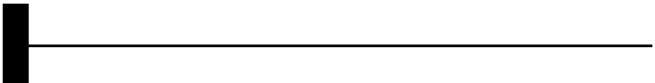
角的程序设计，指出了传统的面向过程与面向对象的程序设计在实际软件工程中的不足与缺陷。直到本世纪初，面向视角的程序设计（Aspect Oriented Programming, AOP）开始在国外得到更多人的关注，其对应的软件工程称之为面向视角的软件工程（Aspect Oriented Software Engineering）。在中国，武汉大学、北京大学、国防科技大学等大学的一些学者在研究。武汉大学以应时教授为核心的研究小组在这方面所从事的研究与探索工作，对笔者产生了深刻影响。现在的问题是：

- ◆ 多视角程序代码的关注点主要是可靠性、安全性、互操作性、可配置性和可管理性。
- ◆ 多视角程序代码必须在已有的程序设计代码上插入、捆绑。
- ◆ 多视角程序代码插入或捆绑到功能程序代码上主要是单方向的与通过工具自动实现的。

我们必须把程序设计方法纳入到系统工程的体系结构和软件体系结构中，建立多关注与多视角的新软件工程学。

世界上还有另一种比面向视角的概念更大的程序设计，其英文缩写同样是 AOP，其对应的软件工程英文缩写也是 AOSE，详细内容请看下一章所介绍的从并发程序设计发展而来的面向代理（主体）的程序设计。

第 9 章



并发程序设计与面向代理(主体)的程序设计

并发程序设计在 20 世纪 80 年代到 90 年代初就已经比较成熟了，尤其表现在当时 DEC 公司的 VMS 操作系统以及后来的 Oracle 公司的系统产品上，它们已经有非常成熟的以进程为中心的系统体系结构的设计了，这从 20 世纪 90 年代初 Oracle 公司的一个系统体系结构（见图 9-1）就可以看出。

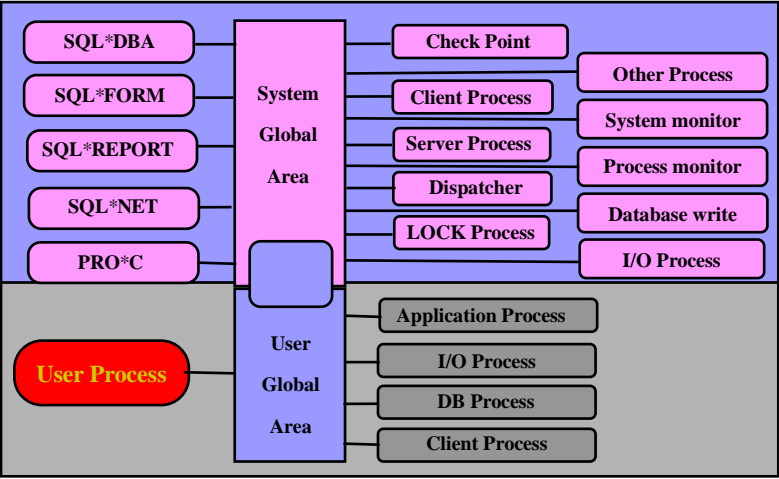


图 9-1 一个系统的进程主体结构

9.1 传统的并发程序设计

传统的并发程序设计是以进程和线程为对象的。并发程序设计从大概念上划分为网络级程序设计、系统级程序设计、进程级程序设计、线程级程序设计以及顺序程序设计和异常处理 5 个环节或阶段。

1. 网络级程序设计

网络级程序设计是在 TCP/IP 基础上利用 Socket、XT1 或者其他设施进行网络服务的程序设计，是在服务器和客户器上进行网络进程（daemon）的程序设计。其设计内容包括：协议设计和协议实现程序设计，具体包括监听、请求、应答、建立连接、结点间通信、撤销连接和 I/O 模式的设计等。网络级并发程序设计的关键是开发面向连接和无连接两种模式的循环服务器和并发服务器。例如，把 TCP 的应用进程创建成一种 daemon 形态的等待进程工作方式，通过一种握手机制把两个进程连接起来。为实现连接操作，定义一个“插口对”（Socket pair）概念。

```
Socket pair = (local_IP_Addr• Local_TCP_Port ,  
Foreign_IP_Addr• Foreign_TCP_Port)
```

各种 TCP 服务器的进程通常处在监听状态，当一个客户器要求服务连接时，服务器收到客户器的连接服务申请后，采用 fork 系统服务产生等待子进程 daemon，然后由 daemon 进程与客户器依照 Socket pair 实现连接，并建立连接状态。

网络级并发程序设计尤其要注意几种不同的 I/O 模式对开发这些服务器的突出意义：

1. 同步 I/O 模式（Synchronous I/O Model）

- ◆ 等待 I/O 模式（Blocking I/O Model）
- ◆ 非等待 I/O 模式（Nonblocking I/O Model）
- ◆ I/O Multiplexing Model（select and poll）
- ◆ 信号驱动 I/O 模式（Signal Driven I/O Model）

2. 异步 I/O 模式（Asynchronous I/O Model）

不同的操作系统，其网络级程序设计的方式与要求是不同的。

网络级程序设计最后还是演变成了进程级程序设计。从图 9-2 中可以看出，网络系统结构是由一系列的进程组成的。

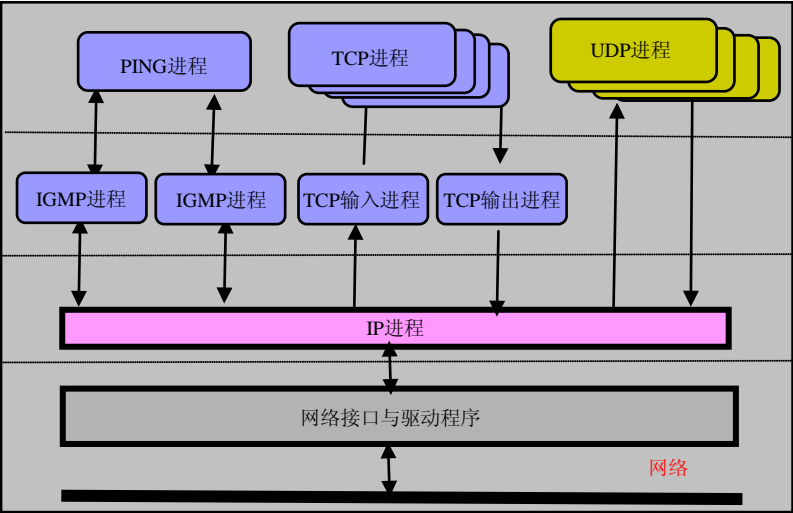


图 9-2 一个网络系统的进程主体结构

2 . 系统级程序设计

系统级程序设计是指在服务器和客户器上对系统的进程和资源进行设计，包括系统进程、系统服务、I/O 设计、驱动程序、系统安装、系统初始化、系统异步自陷、系统管理和系统维护等方面的系统级程序设计（包括存储管理系统、I/O 管理系统、文件系统、命令解释系统、网络系统、系统引导和初始化以及安全系统等方面的设计）。操作系统进入运行状态之后，操作系统的系统进程便成为了主体，这些系统进程包括：空进程（NULL）、系统进程（SYSTEM）、用户交互进程（JOB）、交换进程（SWAPPER）、系统监控进程（System Monitor）、进程监控进程（Process Monitor）以及系统中断处理的辅助进程、系统管理类进程和厂商开发的工具捆绑软件进程等。在操作系统中运行着两类进程：系统进程和

用户应用进程。系统进程作为操作系统的主体运行，而用户应用进程作为用户的主体运行。系统进程和用户进程在操作系统的两个不同空间（系统空间和用户空间）中运行，用户进程运行需要系统进程的帮助与支持。操作系统提供的系统服务也在系统空间内，用户进程调用系统服务是通过软中断机制来实现的。不同操作系统的系统级程序设计的方式与要求是不同的。

现代较大型的应用系统都有相当复杂的系统级程序设计任务，例如，设计一些专门外部设备的驱动程序，修改和新编写系统服务软件，增加应用需要的异常处理程序，增加一些新的安全管理软件，增加新的数据形式和文件形式，设计开机程序和关机程序等。但是，需要说明的是，上述系统级别的软件开发，主要也是在进程概念引导下的程序设计。

3 . 进程级程序设计

进程级程序设计是以进程和进程内的资源为对象进行的，对进程内的资源进行设计，对子进程（包括 daemon）的创建、进程之间的同步和通信进行设计，对进程异步自陷、事件和信号的处理、进程的状态转换以及进程的终止等进行程序设计。

不同操作系统的进程级程序设计的方式与要求是不同的。例如，VMS 操作系统的进程管理的类型表达式的定义为：

```
PMS = VPCB----- 进程控制块（PCB）向量
      × Groups----- 多进程组或多作业组（作业
                        向量）
      × { null, job_controller } 系统空进程和作业控制进程
      × ProcessStateQueue----- 进程状态队列
      × SystemEventQueue----- 系统事件队列
      × LNS----- 逻辑名系统
      × SystemSymbolTable----- 系统符号表
SERVICES
```

```

exe$schedule----- 调度程序
exe$reschedule----- 恢复调度程序
exe$creproc----- 创建进程
exe$crejob----- 创建作业
exe$activate----- 激活进程
exe$hiber----- 进程睡眠
exe$suspd----- 进程挂起
exe$wake----- 进程唤醒
exe$resume----- 进程解挂
exe$pagefault----- 缺页等待
exe$freepagefault----- 缺少自由页面等待
exe$collidedpagewait---- collided 页面等待
exe$reswait ----- 资源等待
exe$wait_lef_and----- 等待多局部事件都发生状态
                        服务
exe$wait_lef_or----- 等待多局部事件中任一发生
                        状态服务
exe$wait_cef_and----- 等待多全局事件都发生状态
                        服务
exe$wait_cef_or ----- 等待多全局事件中任一发生
                        状态服务
exe$delete----- 删除或终止进程服务
exe$inswap ----- 交换入服务
exe$outswap----- 交换出服务
exe$synchnonize----- 同步服务

END
VPCB = (PCB)*----- 进程控制块（PCB）向量
SystemSymbolTable = SymbolTable-- 系统符号表

```

作业组类型表达式

```

Groups = ( Group )*----- 多进程组或多作业组
Group = Jobs ----- 进程或作业组
      x CEF----- 公共事件标志

```

Jobs = (Job)*-----

Job = (process)*-----

x JIB -----

多作业或作业向量

作业

作业信息表

进程类型表达式

Process = PCB-----

x PQB-----

x PHD-----

进程控制块

进程限额块

进程头

又例如 UNIX，UNIX 常用的进程创建方法是采用 UNIXshell 命令注册登录用户，系统可以为用户创建一个控制进程（具有控制诊断的进程）。UNIX 还可以利用系统调用 fork 来创建并发进程，还可以采用 exec 系列函数创建进程。在程序设计中，还可以使用 system 函数创建进程。注意，fork 系统调用所创建的进程是调用 fork 进程的复制进程，复制进程的开始执行点是 fork 系统调用之后，所以 fork 系统调用中没有指明所创建进程的程序映像文件的名称。另外，UNIX 有自己的同步、异步和通信机制。

所谓进程级程序设计，主要是指应用进程的开发。应用系统首先要开发应用的视窗进程，把应用系统所需要的操作与服务划分成同步类型与异步类型两大类，例如编辑操作、文件操作、对象操作和磁盘存储操作。

4．线程级程序设计

线程级程序设计是一种单位更小的进程、子进程意义上的程序设计。线程级程序设计要明确服务是同步还是异步的。如果是异步的，则要采用线程程序设计技术进行设计，尤其是对于一些要求速度快的异步 I/O 服务。其设计内容包括：线程的创建、同步、通信，事件和信号处理，线程的状态转换和线程终止等。线程是操作系统中最小的并发单位，它是组成进程的元素。线程的创建速度比进程的创建速度快 10～100 倍。进程中的所有线程共享如

下内容：进程指令代码、大部分数据、打开的文件、信号处理、当前工作目录、用户和组 ID 等。另外，进程中的每一个线程也有自己独有的内容：线程 ID、寄存器、计数器、栈指针、栈、错误序号、信号屏蔽和优先级等。

5. 顺序程序设计和异常处理

模块级的顺序程序设计主要用于在进程或线程内实现业务服务。在进行上述设计时，应当尽可能地利用系统提供的程序设计资源：类库和动态链接库。异常处理是提高软件可靠性的关键技术。这种异常处理通常是建立在一个进程之内的，基于调用帧的。为了提高软件的可靠性，避免或减少由于异常发生而导致的系统失败，需要在软件开发中进行异常处理设计和编程。也就是说，在每一个程序模块中都有正常处理的流程和异常处理的流程，当软件运行出现异常时，在操作系统的协助下，运行会从正常流程跳到异常流程中进行处理。常规的异常处理包括：继续、信号重发、重新操作、进程内的调用帧的栈跟踪、垃圾箱、撤销操作、拒绝输入、报警和异常出口等。也可以通过进程向系统发出的异步自陷（AST）、状态与事件记录、数据备份等措施强化异常处理。另外，异常处理还包括阻止错误传播和隔离错误等。不进行异常处理程序设计，或者异常处理设计得太简单，对计算机系统来说，稍有异常情况发生，就会死机。

9.2 类型程序设计方法支持并发程序设计

我们可以把进程定义成类型，在并发 PASCAL 语言和 ADA 语言中，早就这么做了。例如，我们可以把进程的抽象数据类型定义为：

```

type Process(Software, Port) is
sorts, Process, Software, Port, State end sorts
constructors
    process_create : Software × Port → Process OR
UNDEFINED
    process_activate: Process → Process
    interruption: Process → Process
    process_reactivation: Process → Process
    exception: Process → Process
    continue: Process → Process
    abortion: Process → UNIVERSE
    process_terminate: Process → UNIVERSE
extensions
    process_get: Process → States
end type

```

进程类型中的港口类型可以如下定义:

```

type Port (Protocol) is
sorts, Port, Port_State, Protocol end sorts
constructors
    port_create: Port_State × Protocol → Port OR
UNDEFINED
    port_open: Port → Port
    port_create_connection: Port → Port
    port_terminate_connection: Port → Port
    port_star_transmit: Port → Port
    port_finish_transmit: Port → Port
    port_close: Port → Port
    port_delete: Port → UNIVERSE
extensions
    port_get: Port → Port_State
end type

```

同样, 我们可以为线程 (thread)、异常 (exception)、驱动程序 (driver) 以及其他并发程序设计需要的机制定义抽象数

据类型。

从上面的介绍中可以看出，传统的并发程序设计不是面向模式与业务领域的，而是面向系统与项目的，还处在原始的状态中。传统的并发程序设计与希望达到的面向代理的程序设计（AOP）与面向代理的软件工程（AOSE）的要求差别还较大，与代理体系的框架、环境、组织、网络、集合和规范多层次的要求相差很远。

9.3 面向代理(主体)的程序设计(Agent-OP)

面向代理的程序设计是在传统的并发程序设计基础上发展起来的新型程序设计技术，而不是以“人工智能”为基础发展而来的技术，它是在计算机操作系统的进程概念的基础上，对更高层的规范代理、代理集合、代理网络、多代理系统组织、多代理网络平台和多代理网络，以代理（主体）为中心进行的程序设计。它建立在面向对象程序设计的基础上，实现面向模式、面向有组织代理群体的程序设计。笔者在多种场合中谈到，如果我国只有面向对象的程序设计技术，而没有面向代理（主体）的程序设计技术，那么我国的软件技术将只有半壁江山。

面向代理的程序设计（Agent Oriented Programming, AOP）是面向模式的程序设计。面向模式的程序设计从应用模式出发，尤其是从分布式并发应用模式出发，依照模式框架和相互关系设计体系结构，模式中体系部件存在的价值全是围绕模式要求的，其特点是始终围绕模式的概念、框架和部件关系按阶段求精、细化，依照这个模式设计概念、功能、结构和实现等。

面向代理的程序设计的更高级阶段是面向有组织群体模式的程序设计。有组织群体模式是指“数字警察大队”、“税务局”、

“商业银行监管组织”和“数字化军”等有组织群体的业务模式。这种应用模式可以是社会性的、协同工作模式的。在设计与实现时，也要紧紧围绕其群体模式实施。如果考虑动态性和可移动性等特性，应当认为群体模式的能力比单一个体系统的能力强大得多、高级得多。这方面最应当注意的成果是各种面向代理的建模方法，其中，OMG 组织的 Agent-UML 比较引人注目。

面向代理的程序设计是基于面向对象的程序设计的。代理与对象是两个既有联系又有区别的概念，不要认为有了代理概念，对象概念就过时了。对象概念是类型概念的实例概念，类型是具有共同性质的事物的集合。类型是这个集合的总称，而对象是这个集合中的一个实体。因此，面向对象的本质是面向类型。有人说，对象是静态的，而代理是动态的，这是不确切的。在对象概念中，尤其是在把对象概念引入到操作系统之后，对象可以是一个进程（例如，在 Windows NT 中，对象概念就包含进程），而进程显然是动态概念。把握对象概念和代理概念，最重要的是从它们的定义出发。在这方面，可以借鉴 FIPA 研究。

代理技术的发展产生了许多基于代理的系统 and 平台。为了更好地开发代理系统，又产生了面向代理的软件工程（Agent Oriented Software Engineering, AOSE），其研究的问题包括代理技术标准化体系建设、开发方法、代理程序设计语言、代理系统的开发平台、代理体系结构和代理通信语言等。代理软件工程包括代理系统工程需求分析、代理系统概念设计规范、代理系统验证技术、面向代理的分析与设计、代理本体的结构、代理部件、代理设计平台、代理运营支撑系统、代理开发工具和代理运营平台等。在《软件行为学》一书中，阐述了一种更为规范的代理程序设计和代理软件工程的方法学和体系结构，提出了代理网格、代理网格平台、多代理、代理网络和规范代理等多层次的系统体系结构，

9.4 类型程序设计支持面向代理的程序设计

```

type Agent (Role,Association,Business,AIB) is
sorts Agent,Agent_State,Role,Association,Business, AIB
end sorts

constructors

    agent_create:Role  $\times$  Association  $\times$  Business  $\times$  AIB $\rightarrow$ 
Agent OR UNDEFINED                                /创立代理/

    agent_activate:Agent  $\rightarrow$  Agent                    /激活代理/

    agent_move:Agent  $\rightarrow$  Agent                        /移动代理/

    interruption:Agent  $\rightarrow$  Agent                      /中断代理执行/

    agent_reactivate:Agent  $\rightarrow$  Agent                  /从中断状态中
重新激活代理/

    exception:Agent  $\rightarrow$  Agent                        /代理运行异常/

    continue:Agent  $\rightarrow$  Agent                          /代理运行继续/

    abort:Agent  $\rightarrow$  UNIVERSAL                        /代理运行异常

终止/

    agent_terminate:Agent  $\rightarrow$  UNIVERSAL              /终止代理的运行/

extensions

    get_state:Agent  $\rightarrow$  Agent_State

```

```

equations for all s : Agent, r : Role, a : Association, b :
Business, aib : AIB let
    get_state(agent_create(r, a, b, aib)) = ( "CREATED" ,
"CREATED" )
    get_state(agent_ activate(s)) = ( "ACTIVATION" ,
"ACTIVATION" )
    get_state(agent_move(s)) = ( "MOVED" , "MOVED" )
    get_state(interruption(s)) = ( "INTERRUPTION" ,
"INTERRUPTION" )
    get_state(agent_reactivate(s)) = ( "ACTIVATION" , —)
    get_state(exception(s)) = ( "EXCEPTION" ,
"EXCEPTION" )
    get_state(continue(s)) = ( "ACTIVATION" , —)
    get_state(agent_terminate(s)) = ( "TERMINATION" ,
"TERMINATION" )
    get_state(abort(s)) = ( "ABORTION" , "ABORTION" )
end type

```

在进程类型定义的基础上，我们还可以定义代理网络类型。代理网络是通过代理的协同港口连接构成的网络。代理网络与代理的状态没有关系，而仅仅与代理的港口有关系。从前面的定义中可以看出，一个代理可以有多个港口。例如，为报文和通告设置的协同港口 **AssPort**、代理之间进行交互的港口 **AgentExPort**、人与代理交互的港口 **HaiPort**、代理实现业务功能的港口 **AppPort**，利用这些代理可以构成不同应用的网络。下面就代理网络的类型进行讨论和定义。

设 X 为代理的对象，设它的港口为 α, β, \dots 。使用代理变量 X 来约束港口的归属，例如，用 $X \bullet \alpha$ 和 $X \bullet \beta$ 表示代理 X 的 α 港口和 β 港口。如果代理 X 的港口集合是 $\Gamma = \{\alpha, \beta, \gamma, \dots\}$ ，则可以采用 $X: \Gamma$ 或 $X: \{\alpha, \beta, \gamma, \dots\}$ 表示代理 X 的港口。

```

type AgentNet (AgentSet , PortAlphabet) is

```

```

sorts AgentNet = AgentSet × Channels × PortAlphabet
    Where Channels = { (x•α, y•β) | x, y : AgentSet and
x•α, y•β : AssPort }, AgentSet, PortAlphabet, BOOLEAN
end sorts

constructors
    emptynet : → AgentNet
    agentnet_inserte : AgentNet × AgentSet → AgentNet
                                                /代理上网/
    agentnet_channel : AgentNet × Channels → AgentNet
                                                /代理建立通道/
    agentnet_register : AgentNet × AgentSet → AgentNet
                                                /代理上网注册/
    [ ] : AgentNet × Agent → AgentSet      /代理/
    ↓1 : AgentNet → AgentSet
    ↓2 : AgentNet → Channels
    ↓3 : AgentNet → PortAlphabet

extensions
    agentnet_create : AgentSet × Channels × PortAlphabet
    → AgentNet OR UNDEFINED                /建立网络/
    agentnet_mask : AgentNet × AgentSet → AgentNet
                                                /屏蔽港口/
    agentnet_mutual : AgentNet × Channels → AgentNet
                                                /内部连接/
    agentnet_reappear : AgentNet × AgentSet → AgentNet
                                                /重现港口/
    agentnet_mutualdelete : AgentNet × Channels → AgentNet
    agentnet_chandelete : AgentNet × Channels → AgentNet
                                                /通道删除/
    agentnet_delete : AgentNet × AgentSet → AgentNet
                                                /代理下网/
    agentnet_deregister : AgentNet × AgentSet → AgentNet
    agentnet_iseq : AgentNet × AgentNet → BOOLEAN
                                                /判断代理网络相
                                                等/

```

```

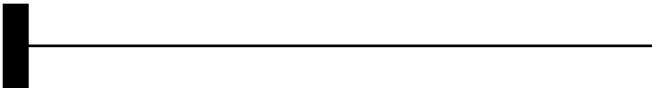
agentnet_isne : AgentSet × AgentNet → BOOLEAN
                                /判断代理网络不
                                相等/
agentnet_isempty : AgentNet → BOOLEAN
                                /判断代理网络为
                                空/
agentnet_isregistered : AgentSet × AgentNet → BOOLEAN
                                /判断代理是否
                                注册/

equations
.....
.....
end type

```

我们要推动面向主体（Subject）或代理（Agent）范畴的软件技术与产品以及系统产业化的软件工厂的建设。关于代理程序设计与代理软件工程，在后面的代理化讨论中还会详细介绍。

第 10 章



可视化与多媒体

可视化是计算结果的理解技术。1987 年 ACM SIGGraph 所做的专题报告《Visualization in Scientific Computing》是划时代的，它是在美国国防部的支持和参与下推出的计算可视化计划。对于科学数据、数据库结构模式、程序结构、数据结构、算法和数学公式的理解，在人工智能进程的长河中，我们才从计算走到数据可视化，即计算→理解→智能化更高阶段中的第二个阶段。可视化技术之所以能够推进成功，一方面是由于当时的计算速度足够快，而对计算打印出的结果的理解与分析要花费大量的时间和精力，对数据的理解技术存在必然的需求。生物学告诉我们，人的大脑细胞的 50% 是与视觉相关联的。另一方面是因为当时的计算机图形学、图像学及技术也相对成熟了。

可视化计划使 IT 进入了 18 个月芯片速度翻一番的时期，人们将其称为计算机的第二次革命。可视化使计算机辅助设计以及仿真与模拟达到了空前美好的境界，虚拟现实得到了应用。可视化的应用领域包括：分子模型、生物学、医学、数学（有限元方法）、地理学、制图学、地质学、气象学、空间探测、天文学、天体物理学、流体力学、城市规划与设计、环境工程、电影制作、军用监控与指挥、实时仿真、信号处理、工程与工业监控、软件工程和程序设计等。视窗是操作系统中最重要的概念——进程以及对象实施可视化的技术。进程、线程以及对象都可以看做窗口。软件对象利用图标来显示。软件的全面可视化是这个时期的特点，几乎所有的软件都是可视化的，例如 Visual C++，Visual Basic，Visual Java，VisualAge，等等。在这个时期，计算机的用户界面从字符终端走向视窗技术支持的图形终端的鼠标点击的应用模式。计算机的可应用性大大提高了。

在计算机技术的发展演变历程中，可视化技术是其第二个里程碑，当时认为：

计算→理解（可视化）→虚拟现实 →

可视化技术的出现，对电影、电视、动画和游戏等技术带来了空前的发展机遇。再加上互联网的支持，视觉内容的 IT 技术达到了空前的活跃程度。如果没有可视化，那么互联网内容建设将失去最重要的部分。

多媒体技术在计算机和网络中也得到了广泛的应用，这里就不介绍了。

第 11 章

光学计算与光学计算机

所谓非线性的计算机体系，包括光学计算机、量子计算机和生物计算机，笔者最看好光学计算机的发展。光学计算机的核心组成包括光存储、光传输、光学显示和光学计算。光学计算技术是光学计算机研制取得突破的关键，因为在光存储、光传输和光学显示等方面已经取得了突破，并且已经获得了广泛的应用。光学计算采用的逻辑系统不是二值逻辑，而是多值逻辑。光学计算机一诞生就与电子计算机存在着根本的不同，电子计算机是一维的，而它是二维或者多维的。光学计算机是面向模式识别的，在人工智能的能力方面，在理论与实际上明显地高于电子计算机。光学计算机的后续研究工作依然十分艰巨，例如光学计算机的程序设计语言、操作系统、运营、管理环境以及工具的开发。在光学计算研究中，原理方面已经取得了突破（例如计算光栅、微型的光学透镜、纳米光学材料技术等），但是依然需要在微型化和工艺方面做更加深入的研究。2003 年，美国网络空间国家安全策略在谈及新技术发展对国家网络空间安全的影响时，首先提到了光学计算和智能代理技术。美国对光学计算机是非常重视的，早在 20 世纪 80 年代中后期就已经开始进行相关研究了。在中国，笔者本人曾在 20 世纪 90 年代初向有关部门提出，要支持光学计算和建立光学计算研究中心，但至今没有得到重视。所以，光学计算机是笔者最为担心和令人鼓舞的计算机技术。之所以担心，是由于光学计算机在基本解决应用问题之后，尤其是在军事方面取得应用进展之后，全世界都会面对“改朝换代”的严重威胁，这种威胁不亚于“原子弹”的威胁；之所以令人鼓舞，是由于在可视化技术取得很大进展的今天，人工智能的研究会有一个新的基础平台，而且是更高层的平台，对人工智能，尤其对解决以视觉为核心的模式识别问题会有很大的帮助。

第 12 章

计算的**理解科学**(形式语义与形式说明)

计算的理解技术包括：程序设计语言、形式语言理论、程序正确性理论、形式语义理论、软件的数学理论、软件工程理论、体系结构理论、计算机图形学、计算机图像学、人际交互理论与技术研究，其中，可视化与多媒体技术为计算机的理解技术发展的一种完满的程度提供了支持。其研究高潮在 20 世纪 70 年代到 90 年代初，在这里，主要向读者介绍形式语言、形式语义和形式说明等方面的内容。

12.1 形式语言理论

形式语言理论（或形式语言学）开创了计算机语言编译程序的科学方法的先河，为计算机高级程序语言的使用奠定了基础。形式语言的文法定义如下：

$$G = (V_T, V_N, P, S)$$

其中， V_T 是终极符的有限集合， V_N 是非终极符的有限集合， P 是生成式有限集合， S 为开始符号，并且 $S \in V_N$ 。 $L(G)$ 称为文法 G 接受的语言。

它充分表现了计算机科学的实用性。这个理论的基础是 Chomsky 文法分类方法：

- ◆ 3 型文法（正则文法） 有限自动机， P 中生成式的形式为： $A \rightarrow \sigma B$ ， $A \rightarrow \sigma$ 。其中， $A, B \in V_N$ ， $\sigma \in V_T$ 。
- ◆ 2 型文法（前后文无关文法） 下推式自动机， P 中生成式的形式为： $\alpha \rightarrow \beta$ 。其中， $\alpha \in V_N$ ， $\beta \in (V_N \cup V_T)^+$ 。
- ◆ 1 型文法（前后文有关文法） 线性界限有限自动机， P 中生成式的形式为： $\alpha \rightarrow \beta$ 。其中， $\alpha, \beta \in (V_N \cup V_T)^+$ ， $|\alpha| \leq |\beta|$ 。 $|\alpha|$ 和 $|\beta|$ 分别表示 α 和 β 的串长度。

- ◆ 0 型文法 Turing 机, P 中生成式的形式为: $\alpha \rightarrow \beta$ 。其中,
 $\alpha \in (V_N \cup V_T)^+$, $\beta \in (V_N \cup V_T)^*$ 。

其实, 上述文法理论是一种字符的项文法理论, 另外, 还存在着图形文法理论, 利用图形的生成式方法描述图形语言的集合构成。

12.2 形式语义学与形式说明

1. 形式语义学

形式语义学是研究形式语言的语义的形式系统。到目前为止, 共有 4 个计算模型 (递归函数论、谓词演算、Turing 机和代数), 于是, 就有 4 个语义流派与之对应:

- ◆ 指称语义学
- ◆ 代数语义学
- ◆ 公理语义学
- ◆ 操作语义学

指称语义学为形式语言的语言成分进行值论域的定义和描述。指称语义学定义了两个域: 语法域 (形式语言系统) 和语义域 (数学域, 已知语义), 是用语义域的对象对语法域的对象进行解释的学问。

一般程序设计形式语言的语法域包括: 语法对象的声明, 用 Dec 表示, 其语义是建立与改变环境; 语言的计算表达式域, 用 Exp 表示, 其语义在于计算产生值; 语言的命令域, 用 Cmd 表示, 其语义改变状态; 还有其他的附加域, 例如 Ide 等。

而语言的语义域通常有: U (环境域)、 E (值域) 和 S (状

态域)。为了解释语言语法对象的语义，需要语义解释函数，例如：

$$D: \text{Dec} \rightarrow U \rightarrow U$$

$$E: \text{Exp} \rightarrow U \rightarrow S \rightarrow E$$

$$C: \text{Cmd} \rightarrow U \rightarrow S \rightarrow S$$

解释语义时，采用递归函数的方法。例如，假定 $\rho: U = \text{Ide} \rightarrow D$, $\sigma: S = L \rightarrow V$ ，可以对语言的循环语句进行语义解释，其形式为：

$$\begin{aligned} C \llbracket \text{while exp do s od} \rrbracket \rho &= \lambda \sigma. (\text{if } E \llbracket \text{exp} \rrbracket \rho \sigma \\ &\text{then } C \llbracket \text{while exp do s od} \rrbracket \rho (C \llbracket s \rrbracket \rho \sigma) \text{ else } \sigma) \end{aligned}$$

在实践中，这些语义学方法需要进行流派融合，产生适合应用需求的语义解释方法。例如，将逻辑方法与代数方法相结合，将指称语义方法与代数方法相结合，构成类型意义上的指称定义，适合于软件自动生成。人们曾经采用指称语义方法为 ADA 语言进行过定义，并产生过实际结果。

代数语义学方法是用代数的方法对形式语言系统进行语义解释的代数形式系统方法，其基本的数学基础是范畴论。一个范畴定义为四元组， $C = (\text{Obj}C, \text{Mor}C, \text{dom}, \text{cod}, o)$ ，其中，与软件有最直接关系的是类别代数范畴。类别代数是一个三元组： $A = (S, \Sigma, E)$ ，其基础概念是 Σ -同构和 Σ -同态，在软件中的应用是 ADT。另外，等式理论和项重写系统是一种代数证明系统，也属于代数语义学研究的范畴。

公理语义学采用形式逻辑理论的方法对程序设计语言或形式语言的语言成分进行逻辑意义上的定义和描述。形式逻辑学包括传统的数理逻辑理论，例如，一阶谓词演算、程序正确性理论、

程序的部分正确和完全正确、不变式、C.A.R.Hoare 的程序部分正确公理系统、E.W.Dijkstra 的程序最弱前置、条件公理语义系统和后来发展起来的模态逻辑、时序逻辑、动态逻辑、多值逻辑和模糊逻辑等各种逻辑学派理论。应当注意，逻辑学的证明是在其数学之外的，是由人来完成的，证明本身不在形式系统之中。所以，数学逻辑方法对于产生一个正确的形式系统和指导人类的逻辑思维活动，是有明显实际意义的。在 20 世纪 80 年代，曾经有人论述过：证明是社会活动。这从另一个侧面告诉我们：这些逻辑学都是给人读的，与计算机（尤其是软件）没有建立起直接的关系。采用这种方法直接指导软件的设计，总有“隔靴搔痒”的感觉。几十年的实践证明，那些企图把数理逻辑的方法简单加以解释，然后搬到计算机和软件问题的研究中的做法，没有产生有实际意义的结果。

操作语义学是以抽象机器为语义解释对象的语义学。这种描述方法曾经产生过许多实际结果，例如，Landin 1964 年提出的 SECD 机器模型为 ALGOL60 进行语义解释；IBM 为 PL/1 语言定义了 VDL 形式说明语言，定义其语义；Knuth 1968 年提出的属性文法，1980 年为 ADA 语言进行语义定义，并自动生成编译程序。

2. 形式说明语言

曾经产生过较大影响的是 20 世纪 80 年代的 VDM（维也纳发展方法，一种指称语义方法）、ITU 的 Z 语言系列（一种逻辑说明语言）和属性文法的说明语言。至于后来提出的语义语言（SL）等，倒没有什么新的东西。

3. 20 世纪 90 年代到本世纪初形式语义学的研究情况

现代各种说明语言中最著名的是 UML 语言，它是一个可视化和半形式化的语言。采用可视化方法是说明语言的一种尝试。

但是，它还没有达得可视化程序设计语言的成熟程度，还有待进一步发展。国内外形式系统理论工作者与信息化要求相一致的研究成果太少了，这方面只有美国国防部提供的某些研究成果值得注意。

第 13 章



系 统 集 成

系统集成标志着一种现代设计、开发、运行和管理的理念。现代系统集成还表现在计算机、硬件、软件、网络与应用系统等方面，包括计算机系统集成、软件系统集成、应用系统集成和网络系统集成等。无论什么系统集成都要反映出多厂商部件供应、多技术体制、多协议与标准、多操作系统平台等必需的选择。系统集成说到底是软件的系统集成。对于软件的系统集成，可以给出如下的定义：

系统集成是定义与引用的实现技术范畴的互操作性技术。

系统集成方面最值得提到的是如下两个成果。

13.1 微软公司的软件集成平台 OLE

微软公司的 Windows 世界闻名，但除了开发软件的技术人员外，很少有人知道微软公司的 OLE/COM/DCOM 计划，它可以说是当时世界上最了不起的计划之一。微软是最早实现公共操作环境的网络信息平台的公司。微软公司的 OLE（Object Linking and Embedding，对象链接与嵌入）计划源于 20 世纪 90 年代初，1991 年发布了 OLE 1.0 版，历时五六年完成了 OLE 的 DCOM 工作（1996 年）。OLE 是面向多厂商、多操作系统平台、多媒体，集计算机、通信与电视于一体，遵循国际标准的网络信息系统平台。它独立于任何一种程序设计语言，并且具有支持交叉操作系统平台的能力，不仅支持同构或异构网络系统互连和互通，还能解决这些系统的互操作问题，实现系统平台上的数据动态交换与程序动态链接。OLE 采用对象化、部件化、可视化与集成化技术，体系完整，便于用户应用、开发与集成，目标明确。通过 10 多年的运行，已经从开始阶段的“文档中心”发展到目前的“网络中心”，实现

了部件的“即插即用”。微软公司的 OLE 连同它的 DDE, NetDDE, COM+, DCOM 和 SDK 等实际上已经成为信息平台的标准了。

OLE 的对象概念包括如下三个：

- ◆ **C++ 的 OOP 的对象概念** 是源程序级别上的、将数据结构与操作封装在一个单位内的对象概念。
- ◆ **OLE/COM/DCOM 的部件对象概念** 部件对象（或简称为部件）建立在二进制可执行标准上，提供指定服务接口的编译后的数据与代码单位。部件是可以被链接和嵌入的可执行对象。
- ◆ **OLE 文档对象概念** 文档对象是组成复合文档所必需的。

OLE 在对象定义方面更接近一般用户的习惯，它把对象定义成一个数据集合及其相应处理信息组成的自封装的软件模块。也就是说，OLE 的对象可以定义成一个三元组：

- ◆ 特性（property）
- ◆ 方法（method）
- ◆ 命令与消息（command & message）

OLE 的部件对象概念（见图 13-1）主要划分为三个层次：部件、对象和接口。OLE 部件是由一个或多个对象所组成的，而每一个对象都是由数据与操作包装的实体，接口是对象为外部部件引用的操作与数据定义的可引用部分，一个部件可以有多个接口。软件部件可以由单个对象组成，也可以由相同类型的对象的集合组成，还可以由不同类型的对象集合组成。

OLE 有如下的特性：

- ◆ **链接（Linking）**

- ◆ 嵌入 (Embedding)
- ◆ 可视化编辑 (Visual Editing)
- ◆ 自动化 (Automation)
- ◆ 拖放 (Drag and Drop)
- ◆ OLE 控件
- ◆ OLE 文档
- ◆ OLE 版本管理

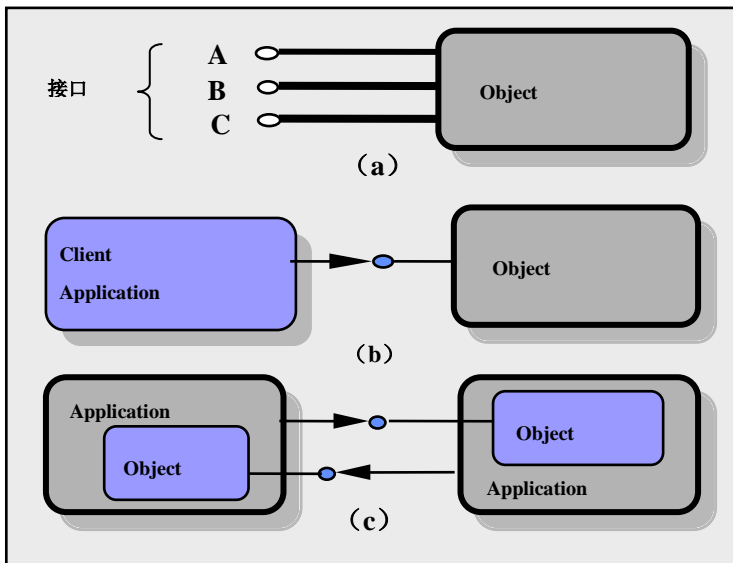


图 13-1 OLE 的部件对象概念

链接是在一个进程中应用系统链接部件对象的机制，通常包括链接容器和嵌入服务器：

- ◆ **链接容器 (container)** 如果一个应用系统从其他应用系统接受被链接对象，那么该应用系统称为容器。

◆ **链接服务器（server）** 如果一个应用系统向其他应用系统提供被链接对象，那么该应用系统称为服务器。

图 13-2、图 13-3 和图 13-4 分别显示了部件对象的进程内、跨进程与跨机器的引用示意图。

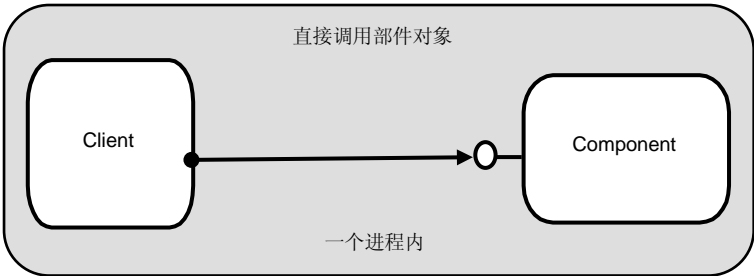


图 13-2 进程内引用部件对象

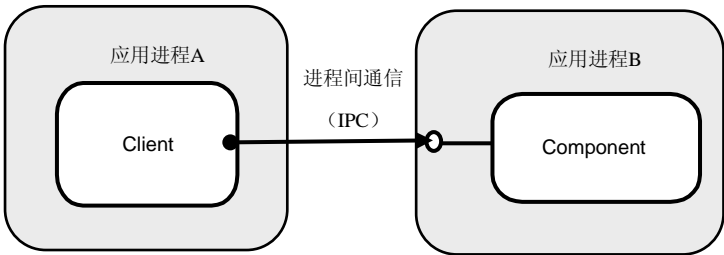


图 13-3 跨进程的部件对象引用通过进程间通信（IPC）实现

嵌入是指进程间的对象链接与引用技术机制，通常包括嵌入容器和嵌入服务器：

- ◆ **嵌入容器** 如果一个应用系统嵌入由其他应用系统产生的对象，那么该应用系统称为嵌入容器。
- ◆ **嵌入服务器** 如果一个应用系统创建的对象提供给其他应用系统，那么该应用系统称为嵌入服务器。

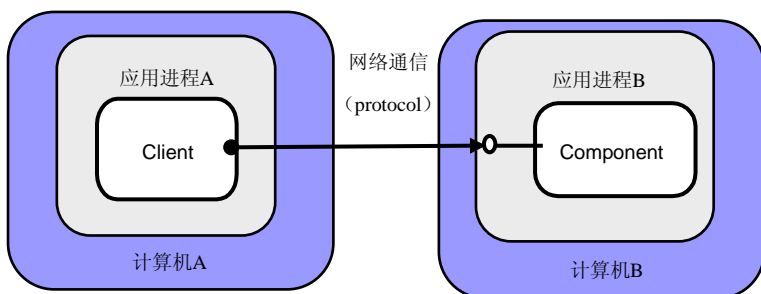


图 13-4 跨机器的部件对象引用通过网络协议实现

13.2 DII COE 段开发技术是更高层次的系统集成技术

大范围的系统集成在美国国防部的 COE 计划中得到了比较好的解决。在公共操作环境上安装部件，部件在平台上以“即插即用”（Plug and Play）的组合方法建设系统，实施“给软件部件穿衣服”的“段”概念的管理体制。“段”（Segment）概念是从用户和管理员的角度定义的，而不是从开发者的角度定义的。段是一个或多个方便管理的功能性软件和数据单位。这种定义对于保留或取消该功能及其相关单位是很方便的。段是管理员可以安装到平台或从平台中取消的最低级部件。分段是一个把部件分解成段和建立段描述符文件的工程过程。

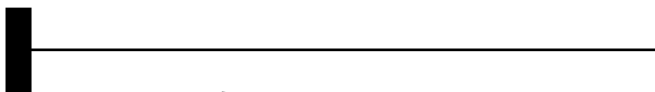
为什么要引入“段”概念呢？段给人的印象是为软件、数据等“穿了一件衣裳”，是对部件按照管理和应用进行重新划分和包装。凡是了解操作系统的人都知道，程序的可执行映像文件都有文件头，文件头的信息包括运营时为程序提供的名称、标识、

存储、局部符号表、全局符号表和程序入口点等，主要是动态链接、调用和访问使用的必要信息。文件头很少为管理和安全设置信息。也就是说，文件头信息基本上是为运营支持服务设计的，很少为管理服务、安全服务和应用服务等提供基础信息。这就好像软件没有“穿衣服”一样。

COE 中引入了特有的段开发的概念和段的开发过程。COE 的开发过程实际上是自动化的集成过程。所谓自动化集成，是指使用自动化工具组合和加载段，按照段的要求进行环境修改，使新加载的段适合于授权用户，并标识一个段与其他段存在冲突的地方。传统系统集成变成了段的加载和测试的初级任务。COE 开发过程划分为如下的几个阶段：段注册、段开发、段递交、段集成、段安装、段认证和 COE 裁剪。另外，这种开发过程完全反映了互联网时代的新型企业模式。DISA 为 DII COE 软件支持活动（SSA）设置了开发者开发段的配置管理（CM）仓库。需要特别说明的是，公共操作环境（COE）的建设，不仅仅是一种标准、结构、基础及软件实现的建设，更是 IT 产业在互联网时代的新型产业模式的建设。

在中国，目前只有中国电子科技集团的华北计算技术研究所（太极计算机公司）掌握这种段开发技术。

第 14 章



统一建模语言（UML） 与可扩展标记语言（XML）

14.1 统一建模语言（UML）

对象管理组织（OMG）是一个世界性的企业联盟式的组织，其成员主要是世界上的 IT 跨国公司，例如 IBM，Microsoft，HP，Oracle，Sun，AT&T，BEA，CA，NEC，Ford，Fujitsu，Ericsson，Unisys，Rational，NASA 和 NTT 等。OMG 是一个非营利机构，主要是使用对象技术对分布式应用集成和开发体系结构进行技术与产品的研究，为标准化提供服务，以实现软件部件的重用性、互操作性与可移植性以及商业适用软件基础作为工作目标，其开发的技术与产品不依赖硬件平台、操作系统和程序设计语言。通过许多年的研究，OMG 已经取得了显著成绩，例如，CORBA（Common Object Request Broker Architecture，通用对象请求代理体系结构）、UML（Unified Modeling Language，统一建模语言）、CWM（Common Warehouse Metamodel，公共仓库元模型）、MOF（Meta-Object Facility，元对象工具）和 XMI（XML Metadata Interchange，XML 元数据交换）等。OMG 机构还有模式驱动的体系结构（Model Driven Architecture，MDA），目的在于解决跨平台应用的兼容性问题，可以说达到了当时的很高水平。OMG 的工作应当值得注意。

在体系结构建模方面，UML 是一个值得关注的体系结构语言。UML 语言是一个可视化与半形式化的模型语言。从软件工程的发展史来看，建立面向对象模型的概念源于 20 世纪 70 年代中期到 80 年代中期，各种面向对象的分析方法和设计方法纷纷问世。20 世纪 90 年代中期之前，面向对象技术的用户纷纷要求建立统一分析与面向对象技术的方法学，在这期间，在新一轮的方法学研究中，Booch'93、OOSE（Object-Oriented Software Engineering，面

向对象的软件工程)、OMT-2 (Object Modeling Technique, 对象建模技术)得到了较多的应用。在这三个模型的基础上,由 Rational Software 公司及其合作伙伴负责,于 1995 年提出了概念,开始了统一建模语言的工作进程。1996 年,Booch, Rumbaugh 和 Jacobson 为 UML 的设计做出了杰出贡献,推出了 UML 0.9; 1997 年,正式推出了 UML 1.0 和 UML 1.1; 2001 年,发展成 UML 1.4; 2003 年, UML 2.0 正式问世。但是,根据笔者判断,OMG 的 MDA 计划还有许多问题需要去解决。我们期待 OMG 的 MDA 取得更大的进步,并期望 OMG 在 UML 2.0 的基础上加强对 Agent UML 的研究,在软件自动生成、软件智能化以及面向代理的程序设计与面向代理的软件工程方面取得具有实际意义的突破。

与 OMG 的 UML 有关系的是分布式计算的中间件技术 CORBA。CORBA 是 OMG 提出的厂商独立、操作系统独立、语言独立、网络基础设施独立的解决信息平台互操作性问题的规范和标准,其核心是对象请求代理(ORB)部件。ORB 是一个机制,利用这个机制,其中的对象可以相互发出请求和接收响应,并且把这种机制跨越在网络之间。世界上有许多 CORBA 产品,有 5, 6 个通用商用 ORB 产品,有几个实时、嵌入 ORB 产品,有 20~30 个自由 ORB 产品、对象事务处理管理者(OTM)、CORBA 部件模式(CCM)和 J2EE/EJB 应用服务器等。全世界有 1000 多个组织使用 CORBA,例如 CNN, American Airlines, Hong Kong Telecom, Charles Schwab, Port of Singapore, McKesson, Boeing, Wells Fargo 和美国国防部及美军等。CORBA 是 Java 的基础。CORBA 的优势包括:已经有 10 多年的历史,厂商独立的中立标准,独立语言及平台,与 J2EE 兼容,多实现,多平台,丰富的服务,数以千计的项目和众多的成功实例。CORBA 3.0 规范开始实施,10 年后将更加成熟、完善。CORBA 的弱点有:较复杂,对于简单问题成本偏高,非时尚,不是大众市

场产品，用户不超过几百万。

14.2 可扩展标记语言 (XML) 与 Web 技术

XML 及其家族语言对互联网技术与应用的发展做出了实际的贡献。XML 的一个主要应用方面是 Web Service。Web Service 方法的焦点在于系统如何互联、互通和互操作，采用如下三个标准：

- ◆ **服务通信协议** 实现网络服务的信息通信采用 XML 语言家族，例如 XML/SOAP (Simple Object Access Protocol, 简单对象访问协议)。
- ◆ **服务描述标准** 为了确认服务的内容及其约束条件，对服务者与服务内容实施鉴别，采用 WSDL (Web Services Description Language, 网络服务描述语言)。WSDL 包括数据类型、接口描述和绑定信息。
- ◆ **服务注册与发现规范** UDDI (Universal Description, Discovery and Integration, 通用描述、发现与集成规范)，用来描述信息 (例如分类、拥有者、业务名和业务类型等)。UDDI 包含指向 WSDL 接口的指针和绑定信息等。

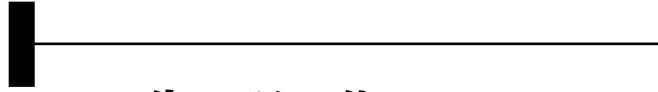
Web Service 有众多的产品，采用统一的标准，而且 W3C 组织在积极强化 SOAP 工作，例如电子业务的 ebXML。Web Service 有许多技术特点，例如 URL (Uniform Resources Locator, 统一资源定位)、WS-Inspection、WS-Security、WS-Interoperability、WS-Referral、WS-Routing 和 WS-License 等。从上面的讨论中可以看出，Web Service 一诞生便是厂商依赖的，而不是厂商独立的。Web Service 的优势是：有 IBM 和 Microsoft 的支持，是大众产品，

并且与语义 Web 和 .Net 等计划协同。Web Service 的弱点是计划刚刚开始,联盟内派系多,承诺过多。尽管如此,我们还是相信 Web Service 将会获得成功。

Web 技术还在科学研究方面得到了重视。科学研究的复杂程度越来越高,科学研究中跨学科的问题越来越多,需要的人力资源越来越多,大量的实验与计算工作也越来越依赖信息化。建立计算机上以及网络环境中科学研究的信息基础设施成为科学研究取得成果的重要条件。在 20 世纪 90 年代中后期,提出了建立 e-科学基础设施的问题。有人提出了网格(Grid)概念。在网格的基础上,英国人提出了语义网格(Semantic Grid)的 e-科学概念。科学研究的代理化是一个发展趋势。在 2002 年的世界数学会年中,著名数学家纳什做了名为《利用代理研究博弈中的合作》的报告,从中可见一斑。W3C 组织专门对语义 Web 进行了研究。语义 Web 是“科学美国计划”中、使用代理技术加强科学研究的发展计划的重要内容。语义 Web 中的重要概念是知识的表示。

美国军方提出了一个将代理技术、XML 和 Web Service 技术结合起来的 DAML(DARPA Agent Markup Language)计划。DAML 是美国国防部使用的代理标记语言,也可以作为数字标签使用(DAML=XML+语义 Web)。语义 Web 是在网页中定义一种语义结构,使网页中的内容对代理和人均可读。定义语义的语言有 DAML, RDF, XML 和 WSDL 等。语义 Web 是当代 Web 中能够使计算机与人均可读的良义定义信息内容的语义方法体系,在 Web 中定义一种数据,使得在各类应用中更有效地发现、自动化、集成和重用。这样,可以发挥与扩充 Web 的能力,达到数据共享、可被自动工具与人处理的更高境界。

第 15 章



代 理 化

15.1 什么是代理技术

在 20 世纪 90 年代初,国内科学界的高层决策人物要求全面推动虚拟现实的研究计划,企图寻找人工智能研究新的突破口。在虚拟现实的研究中,有一些专家更加关注“虚拟主体”世界的研究,这种“虚拟主体”研究主要表现在代理系统的研究上,并且把计算机技术的发展进程演变为如下的发展阶段:

计算 理解(可视化) 虚拟主体系统(代理化) 进一步的虚拟现实

代理 (Agent) 技术诞生于 20 世纪 70 年代,90 年代中后期开始迅速发展,在本世纪得到了更大的重视。这种重视表现在 IT 跨国公司和标准化组织的研究、应用趋势上,从中可以看到一种逼人的态势。首先,代理技术在美国国防部的未来新型战争模式的研究中得到了重视。它认为代理技术在未来的信息战、网络对抗和新型战争模式中将被广泛采用,同时还讨论了代理网格平台建立的可能性和 GIG 计划。代理技术还被美国能源部 (DOE)、美国情报部门 (例如 FBI) 和美国国土安全部等重要机构所采用,它们提出了令人瞩目的发展计划。代理技术同时还是欧洲能源、电力、金融 (银行、证券、保险、信托和租赁等) 等行业风险监管以及互联网内容与行为监管的主流技术。美国、英国科学研究机构正在大规模数据挖掘与采集的科学实验中广泛采用代理技术,寻求代理化帮助。美国国土安全战略将要对国家基础设施信息化实行更加严格的监管与控制,其主流技术是代理技术。其次,世界上多个 IT 跨国公司对代理技术的研究十分感兴趣并积极参与。它们认为,代理技术的发展是最重要的发展,是 21 世纪最重要的软件发展战略。一些世界著名的科学家、技术权威、工业界的领

袖们进行了主题为“什么技术将在未来 5 年时间里改变应用模式、组织结构和行为”的研讨，代理技术得到了极大的重视。一些有远见的科学家认为，在未来 10 年内，操作系统、网络系统、数据库管理系统、智能家电系统、社会服务应用的各类软件系统都要按照代理技术发展的要求重写。许多跨国公司纷纷建立自己的 Agent Factory 产业基地，做了大量的投入，积极迎接代理时代的到来，并以此来摆脱当前的 IT 发展困境。世界 IT 标准化组织已经开发了或者正在开发大量的代理技术标准，这些组织包括 ITU，FIPA，OMG，RFC 和 W3C 等。

对于不同的标准化机构，代理概念的定义不同。下面，我们从几个著名的工业标准来看代理是如何定义的。SNMP 的代理定义为：一组应用进程作为用户的可信代理，这种代理软件体系依据某些标准由管理者、管理代理和操作代理构成，组成协同软件组织关系，共同完成目标任务。这样的标准化代理软件技术简称为 Agent 技术。代理软件可以固定在一个设备上，也可以在网络中的各设备之间移动，寄宿在范围之内或范围之外的设备上，完成指定的任务。OMG 的代理定义为：IT 系统的代理是具有自治性、社会性、交互性、协同性、适应性、代理性、移动性、智能性、可信性和不确定性等特性的组合系统。代理应当是一个可以与环境交互的自治实体。换句话说，代理是可以通过传感器感知环境，并通过受动器向环境执行动作的 IT 系统。研究智能代理的著名组织 FIPA 也有类似的观点。总体来看，代理技术有两种根本不同的研究学派：人工智能学派与计算机科学和信息化科学学派。在代理的人工智能学派中，有一些科学家在研究真正的人工智能意义上的代理模型理论时，把“自治性”（还有人翻译成“自主性”，显然比“自治性”更加智能）作为代理的第一特性。既然把自治性作为第一特性，那么把代理意念、目的、企图、动机、感知环

境等作为研究对象和建立客观、环境、主观、心理的各种观念汇合在一起的模型，便是一种自然的逻辑推理。如此的理论研究是不能指导信息化建设的，即便作为纯理论研究，似乎也仅仅是个构思。有些学者硬要把 Agent 这个英文单词翻译为“智能体”。笔者不赞同把一些有才华的学者引入到这样的代理科学研究中，它也是与大多数年轻科技人员的价值观相悖的。在给学生上课时，笔者讲过，如此的研究也许是人类 100 年或 200 年后的事情，这是一个在电子计算机时代、甚至光学计算机时代都不能完全解决的问题。也就是说，代理技术人工智能特性要符合时代的可行性要求。还有人把 Agent 这个英文单词翻译为“主体”。这些有意规避“代理”这个概念的做法都是徒劳的。

那么，到底什么是代理技术？笔者在《软件行为学》一书中，对代理下了自己的定义：

代理是具有代理性、自治性、社会性、协同性、交互性、移动性与适应性等特性的网络环境内的有组织群体应用进程。

15.2 代理技术的基本特性

在计算机科学和信息化科学学派的代理研究中，有如下一些观点是值得注意的。

1. 代理技术的本质是代理性。

在介绍代理技术时，首先要强调的是代理技术的代理性。所谓代理性，就是代表其他实体的利益与要求来行动。所以我们说，代理技术的第一特性是代理性，其本质也是代理性。代理性是代理的第一特性实际上还是个哲学问题，代理性首先明确了人与代理的关系，强调了人与代理的“主仆关系”。由于强调了代理的

第一特性是代理性，因此，把代理的主观意念、目的和企图等概念留给人，而把代理的行为协同性、行为保密性、行为完整性、行为可信性、行为有效性、内容保密性、内容完整性和内容真实性等作为研究对象就是计算机科学学派研究代理的自然逻辑推理。既然是代理，其授权有限则是当然的。这种代理性主要是指人类在网络中的服务代理，强调人对代理系统负责的特性，但最终还是要进行人工干预的。这也是对那种过分强调代理的自治与人工智能特性的观点的修正。强调代理性，就是不要给予代理过多的不可能完成的决策功能。同时也强调，网络世界中一切虚拟主体的行为最终还是由其“主人”来负责承担的，这也是近几十年中维护网络世界秩序的基本要求。

2. 代理的自治性是标志性特性。

计算机科学学派把代理的自治性作为第二特性来研究。由于强调了代理的代理性是第一特性，所以远离了将代理意念、目的、企图、动机和感知环境等主观概念作为研究对象的模型。所谓自治性，是指无须外部直接干预，便可以基于自己的经验对其内部状态实施某种程度的控制和操作。当一个代理具有一定的摆脱外部控制的独立性时，可以认为这个代理是自治的。代理可以在完全没有直接引用和干涉的情况下操作，这就是自治。因此，自治性是代理必须具备的特性。自治性是代理的第二特性，是代理技术的标志特性。在采用代理技术之前，人与软件是“人与工具”的关系；而在采用代理技术之后，人与软件逐步变成了“主人和奴仆”的关系，这种主仆关系表明了软件根据人的意念自治进行操作和执行任务的深刻含义。对于软件来说，这种从“工具”到“奴仆”的关系是非常有革命意义的发展，是时代的伟大进步。显然，现在没有必要和可能把软件从人类的“奴役”下解放出来。如何确定代理的自治性呢？显然应当与代理的业务需求结合起

来, 代理技术的自治性要符合应用的业务要求。当前, 代理的自治性主要是执行意义的, 而不是决策意义的。

3. 代理的人工智能特性要符合时代的可行性要求。

代理技术的人工智能特性要符合时代的可行性要求, 是对那种过分强调代理的人工智能特性的观点的修正。在冯·诺依曼计算机模型中, 是不可能在人工智能方面有实质性的应用发展的。在现代计算机中, 给予一个确定模型的计算机体系结构过分的人工智能特性, 不符合时代的可行性要求。

4. 代理是面向有组织群体的协同技术。

现代信息网络除发挥传统的共享、交换作用外, 建立信息平台实现网络的协同工作是发展的必然趋势。现代计算机建设强调计算机网络上的群体技术的发展, 这是当代计算机技术发展的重要领域。人们在计算机网络上协同工作的理想变成了现实, 这不仅是由于人类社会是一个密切协作的群体, 而且还得益于计算机网络技术的快速发展。如果网络范围内的应用进程不是无组织的自由个体, 而是一个接受代理委托和具有管理体系、明确分工、固定联系方式、承担共同服务任务目标的应用进程群体, 那么我们称之为多代理群体体系。如果把这种信息搜集扩大到一个大的范围, 例如全中国、几个国家甚至全世界, 那么单一代理就无法胜任了。现实社会的方法是建立一个巨大的人员网络体系来工作。在大范围的环境中, 就必须建立类似人类网络结构的代理网络体系, 由数万、千百万甚至数十亿的代理组成一个巨大的网络虚拟主体体系来完成这个任务, 这种应用系统我们称之为“超级应用”多代理系统。多代理是将代理群体依照一种组织模式构成具有分工、相互协同和实现总体目标的系统体系。大范围系统必须实施合作、协调和分布。协同性需要合作性、交互性与社会性。

5. 建立可生长和可移动的代理群体体系结构是多代理技术追求的建立不确定计算的目标。

建立可生长和可移动的代理群体计算模型是当代代理技术最重要的目标。其一，代理的生长性是指代理系统自身通过克隆、复制等生长技术，繁衍相同类型或其他类型的个体和改变个体的质量，实行系统自身延续和繁殖。这种生长性是可控制的，但是可以达到非常巨大的规模，并可以在大范围网络环境内生长。其二，代理移动性是指将自己从一个环境转移到另一个环境。移动代理也可以理解为在其他计算机上执行可移动代码。从字面上看，移动代理是指在网络上漫游及移动，是动态的。代理的移动性为代理系统的生存性和主动服务特性提供了远程服务的技术基础。这种可生长和可移动的代理群体计算模型是在网络世界中建立的最重要的模型，关于这方面的内容我们会在后面继续讨论。

为了建立这种新型的计算模型，需要建立代理网格、代理网格平台、多代理系统、代理网络、代理集合和规范代理等多层次结构。代理网格理论是研究真正的网络操作系统问题的理论基础。在这里，对代理网格概念做如下描述：

网格是一种框架，将软件和硬件系统融入到这个框架中，其系统将产生原系统所不具备的新能力和新特性（例如调度资源或系统协同的能力和特性），该框架称之为网格。代理网格是多代理系统实现系统协同的公共要求，实现分离代理系统之间协同、创建、合并或联合，构成更大代理系统的协同联合体的框架或机制。

15.3 代理软件工程框架

下面介绍《软件行为学》中的代理技术框架，将概念自上而下划分为如下层次：代理网格、代理网格平台、多代理、代理网

络、代理集合和规范代理。规范代理再往下的概念就是进程。现在分别介绍上述概念的基本含义：

- ◆ **代理网络** 多代理系统、多代理平台和可视化系统的框架。
- ◆ **代理网格平台** 实现更大范围分离多代理系统之间协同、创建、合并或联合的机制。这个平台是多代理系统实现互操作性的环境。
- ◆ **多代理** 在代理网络基础上实现协同机制和业务功能的群体组织或联合体。通俗地说，多代理是一个代理组织。
- ◆ **代理网络** 在代理集合中的代理港口之间建立的通道的集合。
- ◆ **代理集合** 多个不同或相同类型（或角色分工）的规范代理的集合。
- ◆ **规范代理** 在网络环境内具有代理性、自治性、社会性、协同性、交互性、移动性与适应性等特性的有组织群体的应用进程，是构成代理群体必需的代理结构。

在这里，对为什么要做规范代理说几句话。既然要建立多代理组织，就必须将代理规范化。生物世界中的每一个生物都是一种规范化的生物体，没有这种规范就没有物种。这种概念对于网络世界同样适用，我们不能讨论任意进程或代理组织概念。

代理类型用类型表达式定义如下：

Agent = Engine-----	引擎
xBusiness -----	代理应用
xAIB-----	代理行为信息基
Engine = Agent_State-----	代理状态
xAssociation-----	代理之间协同
xRole-----	代理角色
Association = Association_State---	协同状态
xAssPort-----	代理协同港口
xMessage-----	报文传输
xNotification-----	代理通报

×HA_Interaction-----	人-代理交互
×Exchange-----	代理之间信息交换
HA_Interaction = HaiPort-----	人-代理交互港口
×Human_Agent_Interaction ---	人-代理交互
Exchange = AgentExPort-----	代理信息交换港口
×Agent_Exchange-----	代理信息交换
Business = Business_State-----	业务状态
×AppPort-----	代理应用港口
×Function_Applications	
AIB-----	行为信息基

在上述定义中，将代理港口定义为代理的子进程或线程，而代理的协同概念和业务功能概念在本书中也被定义为子进程或线程，其逻辑结构可以用图 15-1 表示。

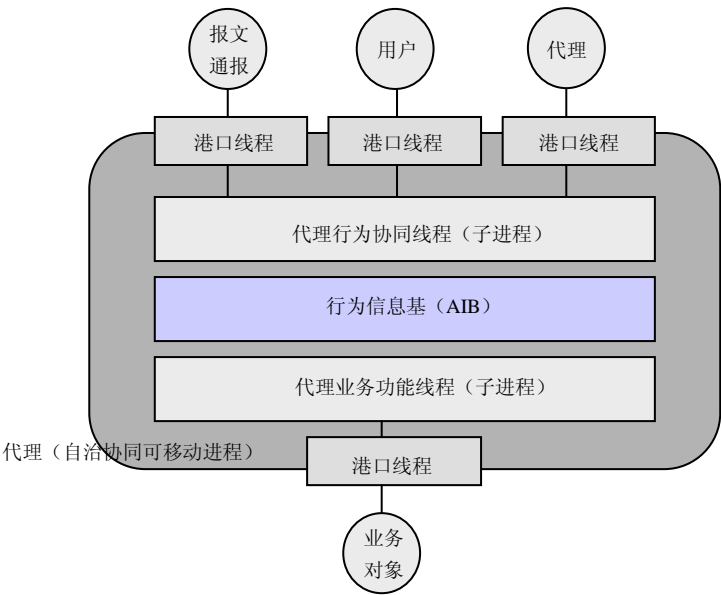


图 15-1 规范代理结构

行为树是进程的“行为踪迹树”。行为信息基（AIB，Action Information Base）是与行为树结构相同、记录主体的行为踪迹及

其行为属性的信息矩阵，如图 15-2 所示。

$$AIB =_{\text{def}} S: \begin{pmatrix} \alpha_{11} & \alpha_{12} & \dots & \alpha_{1n} \\ \alpha_{21} & \alpha_{22} & \dots & \alpha_{2n} \\ & & \dots & \\ \alpha_{m1} & \alpha_{m2} & \dots & \alpha_{mn} \end{pmatrix}$$

图 15-2 规范代理行为信息基

行为信息基（AIB）有两种模式：

1. 行为信息基作为代理行为控制的模式，同时作为研究、分析行为属性的依据。

行为信息基作为代理行为控制的模式，是指行为信息基行为模式中的主体行为踪迹作为代理运行的直接控制方法，依照行为踪迹的路线图，在行为信息基这棵树中行走，并重演行为踪迹指示的行为过程。此时，行为信息基中的行为踪迹还可以作为研究、分析行为属性的依据。

2. 行为信息基不作为代理行为控制的模式，而作为研究、分析行为属性的依据。

行为信息基不作为代理行为控制的模式，是指行为信息基行为模式中的主体行为踪迹不作为代理运行的直接控制方法，代理依照代理控制程序执行的顺序进行运行。应当明确的是，代理控制程序产生的行为踪迹依然可以在行为信息基的路径集合中找到，而不会在行为信息基这棵树中找不到其对应路径。此时，行为信息基中的行为踪迹虽然不控制软件的运行，但是，行为信息基依然是研究、分析行为属性的重要依据，不能缺少。

下面给出行为信息基（AIB）的一个例子：

```
σ : AIB = Subject_Ide----- 主体标识
      × AIB_Table----- 行为信息基表
      × RunModel----- 运行模式（行为踪迹）
      × Agent_State----- 代理状态
α1 : AIB_Table = (AIB_Table_Term)+ - 协同行为信息基表
α : AIB_Table_Term = Action
      ×(AIB_Table +Atom_Action)-- 子行为监管信息基
```

例如，一个单一行为的描述语义内容可以有：

```
a : Action = Action_Ide----- 行为标识
      × Action_Type----- 行为类型
      × Action_Object----- 行为客体标识
      × Action_Input----- 行为输入
      × Action_Output----- 行为输出
      × Action_Attributes---- 行为属性
```

行为信息基控制代理运行的原理如下：

- ◆ 利用行为踪迹的重演来实现代理软件的执行。代理之所以能够按照行为踪迹运行，是由于在可视化系统中记录了人实现某种应用采取的一种或几种操作序列，并能重演这个过程，在代理上实现人工操作序列的“再执行”。
- ◆ 利用操作系统和其他应用软件的资源，代理只有行为信息基（AIB），而行为信息基是一种数据结构的存在格式，没有可执行代码的存在（no code）。行为踪迹的解释执行程序可以选择正规的商品软件。

下面，介绍使用多代理组织的虚拟企业（Virtual_Enterprise）：

```
V_Enterprise = INST MultiAgent(V_EnterpriseAgentNet)
```

```

V-EnterpriseAgentNet=INST AgentNet
                        (V_EnterpriseAgentSet,AssPort)
V_EnterpriseAgentSet = President----- 总经理
                        × CEO----- 首席执行官
                        × CFO----- 财务总监
                        × CTO----- 技术总监
                        × COO----- 市场总监
                        × R&D----- 研发部
                        × Financial_Department- 财务部
                        × Market_Department---- 市场部
                        × Customer_Department-- 销售部
R&D = Manager----- 研发部经理
      × (Senior_Engineer)m----- 高级工程师
      × (Engineer)n----- 工程师
Financial_Department = Manager----- 财务部经理
                        × (Accountant)k----- 会计
                        × (Cashier)l----- 出纳
Market_Department = Manager----- 市场部经理
                        × (Analyser)q----- 市场分析员
                        × (Assistant)r----- 项目经理
Customer_Department = Manager----- 销售部经理
                        × (Sales)z----- 销售人员
    
```

从代理的逻辑结构中可以看出，代理功能有时是很复杂的，这样的代理称为“胖代理”。因为每一个代理都有自己独立的应用资源，所以“胖代理”移动起来很困难。于是，在平台的实际开发中，要求把“胖代理”化解为“瘦代理”结构，把代理划分成引擎与应用两部分。引擎是代理的基本部分，通常比较“瘦”，便于移动；而代理的应用部分通常是资源，不便移动。把这些资源配置在系统中的每一台计算机上，以便无论代理的引擎部分走到哪一台计算机上，都有相同的应用资源可以利用。也可以把代理的应用资源存储在局域网或区域网中的专门计算机上，而在网

络范围内的每一台计算机上直接运行代理或多代理引擎（瘦代理），共享网络中的统一应用资源。显然，这种方式只适合于局域网和区域网范围内代理化的系统。由于多种代理的存在，产生了多种技术标准和代理之间的互操作性问题，于是，提出了统一建立代理运营、管理、监管和认证平台的要求，为各种代理能够运行在统一的平台上提供服务和支持。平台根据不同应用类型分为运营平台、管理平台、监管平台和认证平台等。在对抗类代理或多代理系统中，作战区在对方领地，不能由平台提供应用资源的支持，而是由后勤支援代理提供应用资源的支持。

从上面的虚拟企业的类型表达式的定义中可以看出，这个企业最终需要如下的虚拟人员，这些虚拟人员都是不同业务功能的代理：

```

President=INSTEnterprise_Agent
            ("PRESIDENT",President_Business,E_AIB)
CEO = INST Enterprise_Agent ("CEO", CEO_Business,E_AIB)
CFO = INST Enterprise_Agent ("CFO", CFO_Business,E_AIB)
CTO = INST Enterprise_Agent ("CTO", CTO_Business,E_AIB)
COO = INST Enterprise_Agent ("COO", COO_Business,E_AIB)
Manager = INST Enterprise_Agent
            ("MANAGER",Manager_Business,E_AIB)
Senior_Engineer=INSTEnterprise_Agent
            ("SENIOR_ENGINEER",
            Senior_Engineer_Business, E_AIB)
Engineer = INST Enterprise_Agent
            ("ENGINEER",Engineer_Business,E_AIB)
Accountant=INST Enterprise_Agent
            ("ACCOUNTANT", Accountant_Business , E_AIB)
Cashier = INST Enterprise_Agent
            ("CASHIER",Cashier_Business,E_AIB)
Analyser = INST Enterprise_Agent

```



```

        ("PRESIDENT",Analyser_Business,E_AIB)
Assistant = INST Enterprise_Agent
        ("PRESIDENT",Assistant_Business,E_AIB)
Sales = INST Enterprise_Agent
        ("SALES",Sales_Business,E_AIB)

```

代理管理平台通常具有代理生存期管理、代理身份管理、代理安全管理、代理移动管理、领地范围管理与维护等功能。

代理技术的概念结构也可以形式化描述，例如：

ag:	AgentGrid-----	代理网格
ma:	MultiAgents-----	多代理
mapf:	MultiAgentPlatform-----	多代理平台
vs:	VisualizationSystem-----	可视化系统
io:	Interoperability-----	互操作性
pb:	Plat Business-----	平台业务
ps:	Plat State-----	平台状态
an:	AgentNet -----	代理网络
mas:	MultiAssociation-----	多代理协同
mb:	MultiBusiness-----	多代理业务
ms:	MultiState-----	多代理状态
as:	AgentSet -----	代理集合
ch:	Channels -----	代理通道
a:	Agent -----	规范代理

代理、代理集合、代理网络、多代理系统、代理平台和代理网络的形式化定义如下：

```

ag = λ(ma, mapf, vs)•Frame(ma, mapf, vs)
mapf = λ(ma)• AgentEnv(ma, io, pb, ps)
ma = λ(an)• Organization(an, mas,mb,ms)
an: AgentNet = AgentSet ×Channels
as: AgentSet = {x | x:Agent }
ch: Channels = { (x•α, y•β) | x, y:Agent and x•α is a port of the agent
                x and y•β is a port of the agent y}
a: Agent

```

其中，

```
Frame: MultiAgents×AgentPtatform×VisualizationSystem→
AgentGride,
AgentEnv: MultiAgents×Interoperability×Business×State→
AgentGridePtatform,
Organization: AgentNet×Association×Business×State
→ MultiAgent
```

是一些函子。

总之，代理技术，尤其是多代理技术的发展,使软件不仅仅作为工具为人类所使用,而且成为代表人类在网络世界中自治活动的虚拟主体,并能联合起来构成有组织群体为其“主人”服务。代理技术是计算机科学与技术中继计算技术、网络技术、对象技术和可视化技术之后,带动IT全局发展的更高层面的新技术,它为IT发展带来一个新的时代:代理化时代。读者可以从后续讨论中看出代理技术在当代和未来信息化中所发挥的作用,自己做出评价。代理技术框架或代理软件工程框架如图15.3所示。

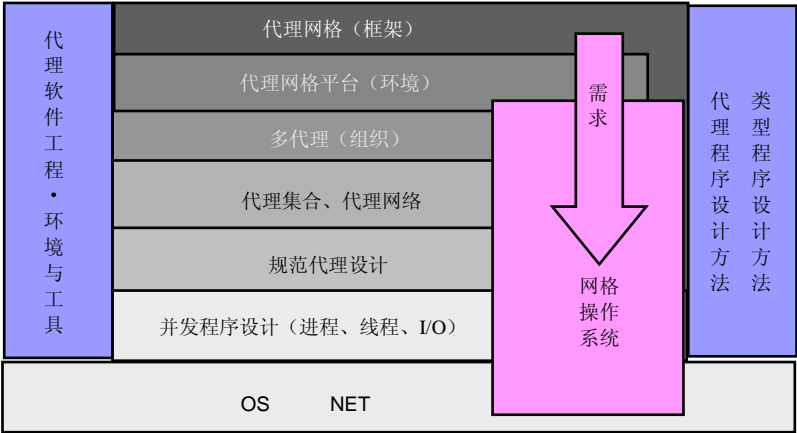


图 15-3 代理技术框架结构

15.4 代理技术工厂

下面,我们来介绍 Agent Factory 的概念。在面向对象技术和部件技术中,曾经有过对象工厂(Object Factory)、类别工厂(Class Factory)和类型工厂(Type Factory)自动化技术,它们是系统生成、运行和应用的新型技术模式。在面向代理的程序设计技术中,也有一种自动化技术,为代理产生、运行、管理和应用提供自动化技术支持。请注意,下面谈到的 Agent Factory 不是一个技术实现的名称,而是真正的产业概念上的工厂。

1. 什么是 Agent Factory

在面向模式和面向对象的程序设计(OOP)基础上,实现面向代理的程序设计(AOP)和面向代理的软件工程(AOSE)方法。针对代理网格、代理网格平台、多代理、代理网络、代理集合、规范代理和应用进程等不同并发和分布层次, Agent Factory 是实现主体(Subject)或代理(Agent)范畴中的系统框架、环境、组织、网络、集合、规范、进程、线程和功能部件的设计、开发、测试和生产的软件产品与系统的工厂。简单地说, Agent Factory 是实现面向主体或代理范畴的软件产品与系统的工厂。

2. 为什么要建设 Agent Factory

从产业角度看,我国面向主体或代理范畴的软件技术、产品以及系统的开发,基本上是个空白。再不抓住机遇,我国软件产业将会在重大技术领域(关系到国家安全、基础设施信息化、电子政务、电子商务与人民生活信息化推进的大问题)出现时代性

的落后。Agent Factory 是当代软件业最高水准的高技术产业。另外，代理技术应用有巨大的市场需求，例如：

- ◆ 多代理群体有组织协同计算的主动服务系统（Agent MIS），20 多年来开发的 MIS 系统逐步实现代理化升级与换代的需求。
- ◆ 公众服务的个人代理助理软件系统的需求。
- ◆ 在大范围的环境中，超海量的网络虚拟主体和高智能体系的“超级应用”的需求。
- ◆ 建立国家基础设施和政府监管现代化体系的需求。
- ◆ 建立公众服务信息化和监管、认证、保障现代化体系的需求。
- ◆ 建立网络世界的数字军队、数字警察和安全部队的需求。
- ◆ 我国信息化安全建设的广泛需求。
- ◆ 其他。

3. 如何建设我国的 Agent Factory 产业

根据各业务领域应用代理技术的特点，可以建设不同领域的 Agent Factory 产业，其方案内容如下：

- ◆ 建立 Agent Factory 项目推进的总体单位。
- ◆ 建立面向电信与广电的信息 Agent Factory。例如，为电信与广电领域的大规模数据与信息挖掘、检索、过滤和发现服务，为电信监管、认证以及公众服务信息化安全服务需要的代理技术提供产品与系统。
- ◆ 建立面向电力、交通、民航、铁路、能源和环境保护等领域测控系统的 Agent Factory。例如，为国家智能电力网、供水、供气、环境检测、煤矿安全检测网络以及公众服务代理网络系统、家电网络系统和各类传感器的代理网络平台提供服务。

- ◆ 建立面向金融、税收、财政和工商等经济领域的 Agent Factory。例如，为这些领域的监管现代化建设提供代理技术产品。
- ◆ 建立面向军事与政法领域的 Agent Factory。例如，为国家建立数字军队、数字警察和情报代理等提供服务。
- ◆ 建立面向 MIS 系统代理化的升级与换代以及互联网范围内服务业务的 Agent Factory。例如，建立面向个人代理助理软件系统的 Agent Factory。
- ◆ 建立面向信息化安全建设的 Agent Factory。例如，为可信网络平台建设代理技术产品工厂。
- ◆ 建立 Agent Factory 实验室等。

第 16 章

互操作性平台理论

16.1 互操作性概念

互操作性是当代信息化中最重要的概念之一，是指两个或多个系统或部件交换和使用信息的能力（IEEE STD 610.2）。信息化在早期是不考虑互操作性的，20 多年的信息化建设，建成了数百万、数千万个标准，千家产品，烟囱林立、相互割据的信息系统，严重制约着互联、互通和互操作问题的解决。有人认为，系统之间的互联、互通和互操作可以通过统一产品与系统的接口标准来实现。也就是说，并没有独立的互操作性概念。互操作性概念的发展与演变，经历了“信息共享”、“信息交换”、“数据共享工程”、“信息系统互联、互通和互操作的技术”、“互操作性平台”、“系统的系统”、“信息交易”和“信息化总体方法学概念中的互操作性技术”等认识过程。对互操作性问题的理解与认识在业界中是不同的，例如，有人把互操作性理解为可移植性、兼容性、平台独立性、语言独立性、资源重用性、系统与设备的一致性等。这是由于不同企业对互操作性的关注点不同，提出的解决问题的方案也不同的缘故。互操作性之所以被当代信息化建设关注，是由于当代信息化关注的是发挥综合效益，强调综合性的运营、管理和服务，例如建立公共操作环境、公共办公环境、公共交换环境和公共交易环境等。互操作性强调面向众多系统的平台建设，提倡独立平台技术与产品开发，强调信息化建设的更低成本和资源效益的充分发挥。在互操作性方面，经过了较长的认识过程，有许多教训与经验。在互操作性平台建设方面，取得了许多成绩，其中最显著的是如下几个方面：

◆ Microsoft 公司的 OLE/COM/DCOM 体系

- ◆ 美国国防部的公共操作环境（COE）
- ◆ OMG 的 CORBA 体系
- ◆ 基于 Web 服务的互联网体系

在下面的内容中，我们将重点介绍美国国防部的公共操作环境（COE）计划。

公共操作环境（COE）技术源于美国国防部的 DII COE 计划。通过 20 世纪 90 年代初的海湾战争，美军发现联合作战体系存在巨大缝隙，高技术战争成本之高难于承受，需要大幅度降低作战成本（尤其是降低 IT 成本）。冷战后，为了满足全球联合作战的需要，改善作战决策能力，实现三军联合作战指挥，克服各军种/兵种的信息系统难于互联、互通和互操作的弊端，消除信息系统“烟囱林立”和信息孤岛的现象，美国国防部推出了公共操作环境的网络信息平台计划。在制定统一平台发展计划的同时，还要制定长期服务的成熟技术发展战略。这个成熟发展战略明确地对 IT 的所谓“18 个月发展周期律”说“不”，从根本上挤压了当时 IT 发展的泡沫，同时也说明了由可视化技术应用开始、由网络应用扩展推动的 IT 发展的动力在衰竭，应当探求新的应用模式的技术发展动力。美国国防部在 1992 年开始启动 DOD COE 计划，目的在于实现美军 C4ISR 的公共操作环境的体系结构设计。1998 年到 2000 年期间，美国陆军、海军和空军对 C4ISR AF V2.0 进行适当修改后，确立了各自的体系结构标准。2000 年后，北约组织（North Atlantic Treaty Organization, NATO）也基于 C4ISR AF V2.0 的改写版本，提出了其体系结构标准和互操作性管理计划。从上世纪末、本世纪初开始，美国国防部又在酝酿全球信息网格体系（GIG）。从现在的初步发展计划来看，发展目标没有 DII COE 计划明确，要解决的问题也不是十分清楚，基本上是在 DII COE

计划的基础之上开展某些新的研究。我们应当密切注意这个计划的实施。

美国国防部的公共操作环境网络信息平台技术采用了如下一些标准和规范概念：

- ◆ POSIX (Portable Operating System for Information Exchange, 信息交换可移植操作系统)
- ◆ DOD TAFIM (Technical Architecture Framework for Information Management, 信息管理技术体系结构框架)
- ◆ DOD JTA (Joint Technical Architecture, 联合技术体系结构)
- ◆ DOD UIS (User Interface Specification, 用户接口规范)
- ◆ DII COE (Common Operating Environment, 公共操作环境)

公共操作环境的定义如下：

公共操作环境是标准、规范、指南、体系结构定义、软件基础设施、重用部件、API、方法、运行环境定义、引用实现和系统建立环境的方法的集合。

DII COE 计划比 OLE 做出了更大的贡献。COE 概念可以简单归纳如下：

- ◆ COE 是建设应用系统的基础。
- ◆ COE 是开放的系统结构。
- ◆ COE 是引用的实现。
- ◆ COE 是实现策略。
- ◆ COE 是互联网时代的新型产业模式。

互操作性长期被认为是技术标准和技术实现的问题，其理论问题很少被考虑，例如，什么是互操作性问题。在这里，笔者给出一个互操作性理论描述的定义：

互操作性平台理论是站在信息体系结构的角度，以全局和多层次视点考察定义（Definition）与引用（Reference）的关系的理论，要研究定义与引用的规范与实现的全部含义。其中，系统集成主要是定义与引用的实现技术范畴的互操作性技术。

根据这个定义，互操作性是研究信息系统的定义与引用的关系理论、规范和实现方法与技术的。我们知道，对互操作性的理论问题是缺少研究的。例如，有如下的错误做法：为了解决互操作性问题，建立一种网络模型或网络自动机模型，企图在这种模型上研究互操作性。不同的自动机（例如，具有不同字符集、状态集合和操作集合等）在这个网络中还要引入通信或信息交换的协议。显然，这样的互操作性研究并没有引起计算机科学家的关注，因为他们认为这是技术问题，而不是理论问题。从顶层概念设计（总体学角度）开始，就应当把互操作性作为一个重要的关注点，在运营体系结构、系统体系结构和技术体系结构等环节中予以关注。对于互操作性的高层概念，我们将在后面的信息化总体学与体系结构的讨论中做重点介绍。下面，主要介绍互操作性的如下内容：

- ◆ 互操作性的实现技术
- ◆ 互操作性带来的服务模式的改变
- ◆ 互操作性引发的安全问题
- ◆ 互操作性的测评认证问题

下面分别介绍这些相关内容。

16.2 互操作性的实现技术

互操作性的实现技术实际上主要是“引用实现”（reference

implementation) 技术。引用实现技术包括以下几种：

进程内的引用实现技术

进程内的引用实现技术可以划分为静态链接和动态链接引用实现技术（图 16-1）。链接技术的提出是由于现代计算机体系结构实行分层（操作系统划分为核心层、执行层、管理层和用户层，层间通过特权改变管理模式的指令相互进入和访问，层间的软件不能直接引用）、分区（虚存管理，不同用户进程使用相互隔离的存储空间）、分权（对系统主体划分角色和使用系统资源的权限）管理体系而引起的。静态链接是由多种语言编写的程序以及新编写的程序模块与可以共享的软件模块通过链接程序（Linker）链接在一起，建立固定关系的引用实现机制。进程内的静态引用机制主要包括函数或过程的直接调用，操作系统服务通过改变管理模式的软中断指令的方式为应用进程提供引用实现服务。这种进程内的静态引用机制也可以通过进程与自己范围内的线程以及进程内线程间的资源共享机制，提供不同线程和线程与进程之间的引用服务。

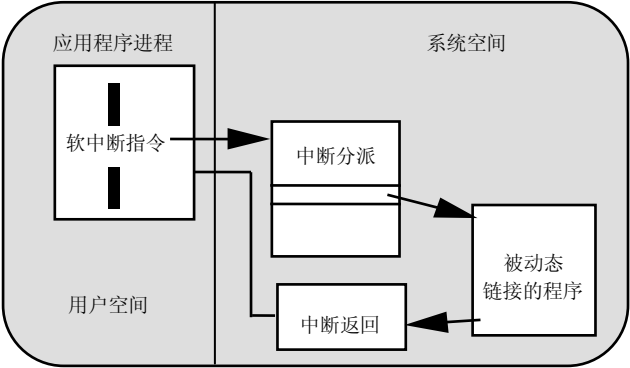


图 16-1 应用进程引用系统服务

进程内的动态引用机制主要是指应可视化应用要求产生的引用实现模式，如图 16-2 所示。由于通过菜单和鼠标进行选择，可视化应用中对于服务的选择是在进程运行过程中进行的，当然就不可能在编译和链接过程中确定下来。在软件设计中，只能为可视化应用提供选择范围，在技术实现上通过函数或服务的分派表格来确定选择的服务，并将其与实际服务程序联系起来（例如 Microsoft OLE/COM 的 VTBL 向量表，如图 16-3 所示）。在实际应用中通过窗口选择，一旦确定选择，便可以产生确定的服务名称，从而确定服务过程的引用关系。这种动态的引用模式既可以用于系统服务的选择，也可以用于动态创建进程内的线程。

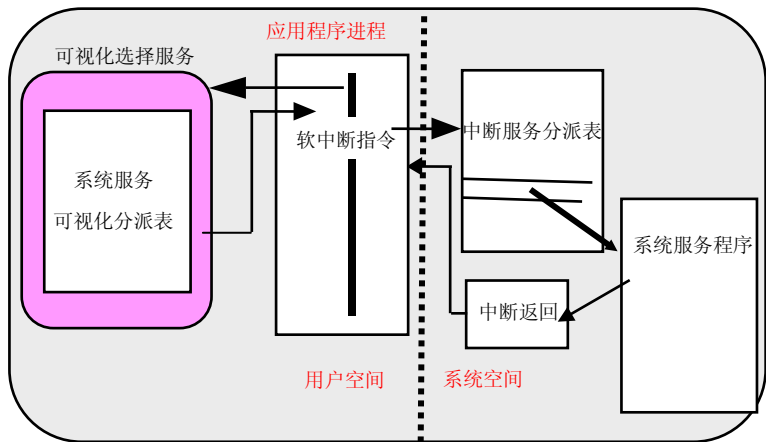


图 16-2 应可视化应用要求产生的引用实现模式

进程间的引用实现技术

由于现代操作系统把进程包装成相互隔离的实体，它们之间不可能实现进程内的引用，更不可能直接进行访问，所以进程间的引用关系是在进程运行起来后建立的。这种进程间的引用关系的确立是通过进程间通信（IPC）实现的。这种进程间的引用关系

为计算机间和网络间的引用关系奠定了实现基础。

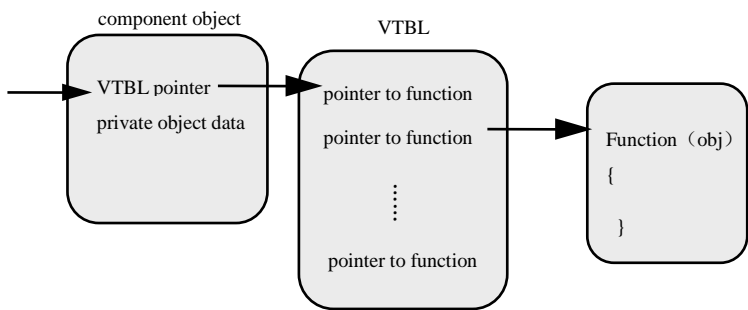


图 16-3 Microsoft OLE 的 VTBL 向量表

跨计算机的网络内引用实现技术

跨计算机首先是跨操作系统，这种引用实现是两个操作系统管辖下的进程引用实现。跨操作系统可能意味着跨不同技术体制的操作系统（例如 Windows，UNIX 或 VMS 等），但是可以在一个网络技术体制之内，即使用相同的网络通信协议。实现的方法包括：网络虚拟计算机（例如并行虚拟机）和代理计算机（一种中间件技术，例如 OMG 的 CORBA）。两个系统之间不能互联、互通和互操作，但是通过一个中间件，通过某种转换，即可完成互联、互通和互操作，OMG 组织的 CORBA 为我们提供了借鉴与启示。

跨网络的引用实现技术

这种引用实现是指多个网络和多个网络技术体制之间的引用实现机制，除了跨计算机的引用技术之外，还要考虑不同的网络技术体制之间的引用技术问题。

开放程序的源代码和接口的源代码技术

通过开放源代码解决互操作性问题也是一条途径。开放也得

有方法，Web Service 给我们提供了启示。这种方案首先是跨国公司的方案。

COE 段开发的引用实现技术（在系统集成中介绍过了）
代理主体的引用实现技术（参见图 16-4）

采用代理技术解决互操作性问题实际上是由于两个主系统之间不能互联、互通和互操作，但是各自的主系统都有各自的代理服务系统，而它们的代理服务系统可以互联、互通和互操作。也就是说，通过主系统之间实现标准化来解决互操作性问题，代价高，受到的限制大，难于实现；双方通过建立辅助系统，专门解决互操作性问题，不牵涉主系统，不影响主系统的应用，代价低。制定代理服务系统的互操作性标准，建立互操作性的代理服务系统。使用代理技术解决互操作性问题是近些年提出的方法，受到了企业与用户的普遍欢迎。本书中介绍的计算网格平台采用的就是这种技术，这是解决应用系统互操作性问题的低成本出路，也是互操作性的保障技术，是一种更有前途的技术。

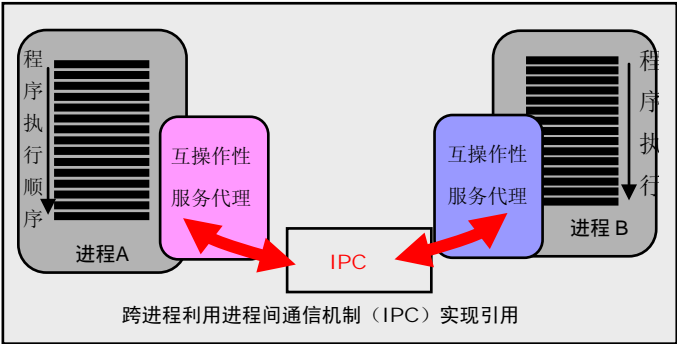


图 16-4 跨进程引用实现机制

16.3 互操作性带来的服务模式的改变

在这方面取得成功的案例是美国国防部制定的 DII COE 计划的互操作性和一致性测评认证，这项工作是通过一个称为软件支持活动配置管理的服务结构（SSA-CM）完成的。信息技术厂商如果要为美国军方开发信息技术产品，需要向美国国防部申报，经批准并获得生产许可证和密码口令之后，就可以从网上下载 COE 的生存期各阶段的标准、规范、文档、工具和核心软件实现了。厂商在进行规定范围之内的开发后，递交美国国防部 COE 测评机构进行测评，测评机构给出产品安全性和互操作性的级别评估。这种测评方法可以把测评服务大大前移，企业从设计阶段开始到后期各阶段都可以得到测评认证的服务。从总的测评时间来看没有变化，但是，从产品最后得到认证的时间来看，大大前移了。由于在设计阶段测评工作就介入了，所以测评的质量和水平也得到了提高。同时，还可以评估信息技术厂商的能力成熟度模型（CMM）。目前，为美国国防部 COE 开发产品的有几千家厂商。

16.4 互操作性引发的安全问题

互操作性带来的安全问题与我们经常谈到的网络安全问题和系统安全问题是不同的，这些安全问题不是每一个单一系统的技术机理、技术体制、工程实施、运营管理体制造成的。多系统融合会造成多种技术体制、工程实施、运营管理体制的新的冲突与矛盾，这种新的矛盾与冲突必然为多系统融合或互操作性带来新的安全问题。这一点对国家基础设施信息化、电子政务、电子商

务和公共服务体系非常重要。

有如下的一个普遍结论：

“由多系统融合或互操作性产生的安全问题要远远大于各单一系统安全问题的总和”。

互操作性引发的安全问题通常包括如下方面：

- ◆ 由于多系统对安全问题的要求不同（访问控制原则、授权原则、管理原则、审计原则、防护原则、监控原则、安全策略等要求不同），在实现多系统的融合或互操作性时，首要的问题就是实现系统之间的隔离、系统标识、鉴别和认证。请读者注意，这种系统隔离与网络隔离是有本质区别的。
- ◆ 由于多系统的用户、主体和客体要求实现不同的安全原则，因此，不仅要实现系统隔离，还要实现用户、主体和客体等方面的隔离、标识、鉴别和认证，在隔离、标识、鉴别、认证、监管、管理和控制中实现互操作性。
- ◆ 多系统融合扩大了系统的网络环境范围和规模问题，这种大范围和大规模引发了新的安全问题，尤其对技术发展提出了更高的要求。
- ◆ 由于多系统，尤其是公共服务信息化系统建设，考虑安全的基本出发点建立在可信与不可信混合工作状态基础之上，基本原则建立在怀疑和不可信基础之上，因此，对于可信管理、可信监管、可信控制和可信认证提出了更高的要求，对于问题与事件的发生定位与追踪提出了更高的要求。
- ◆ 多系统引发了不同用户终端的连接，产生了严重的用户可信接入的问题。

16.5 互操作性的测评认证问题

在 COE 的建设中, 测评认证是非常重要的。COE 建设中强调一致性(即插即用)、互操作性、安全性和重用性, 这些特性的落实都是通过测评认证实现的。其中, 一致性、互操作性和安全性测评是 COE 建设的关键内容。在 DII COE 中, 规定了 COE 的一致性和互操作性概念。一致性是一个整数, 用来表示一致性的级别, 测量如下 4 个方面的特性:

- ◆ 度量段或系统达到 COE 描述的规则、标准和规范的一致性。
- ◆ 度量段或系统适合于 DII COE 引用实现集成的一致性。
- ◆ 度量段或系统使用 COE 服务和共享数据的一致性。
- ◆ 度量段或系统的标准和规范。

一致性划分为 4 个范畴: 运行环境(Runtime Environment)、风格指南(Style Guide)、结构兼容性(Architectural Compatibility)和软件质量(Software Quality)。每一个范畴又划分为若干级别。运行环境划分为 8 个级别, 其中, Level 1 到 Level 3 是 COE 接口标准, 不是真正的系统集成标准级别; Level 4 到 Level 8 是真正的系统集成标准级别。

对于互操作性测评, LISI 提供了另一种方法, 称为 LISI 互操作性成熟模型, 它与 DII COE 互操作性的一致性的 8 个级别的定义不同。LISI 对互操作性的认证和评估以互操作性成熟模型、引用模型、能力模型和实现选择表为基础, 最终产生 LISI 认证产品: 互操作性框架、互操作性度量、比较表和结构产品。LISI 互操作性成熟模型定义了 5 个级别:

- ◆ Level 1 (Isolated Systems) 人工网关，实行磁盘、磁带的媒体交换，无线路连接。
- ◆ Level 2 (Connected Systems) 同类信息交换，例如电子邮件、字符文件，线路实施物理连接，数据和应用分离。
- ◆ Level 3 (Distributed Systems) 异类信息交换，能实施基本的合作和综合业务，具有最小的公共功能，数据和应用分离。
- ◆ Level 4 (Integrated Systems) 共享数据库，复杂合作，实施公共操作特性，共享数据，应用分离。
- ◆ Level 5 (Universal Systems) 跨领域的信息和应用共享，高级合作，能交互地进行公共操作特性修改，事件触发全局数据库修改。

另外，在北约组织（NATO）为了实现统一指挥，在北约组织 C3 的技术体系结构中定义了一个互操作性引用模型（Reference Model for Interoperability, RMI）NC3TA，它也有与 LISI 类似的结构：

- ◆ Level 1 (无数据交换 , No Data Exchange) 没有物理连接存在。
- ◆ Level 2(无结构数据交换 , Unstructured Data Exchange) 人工可互操作交换无结构数据（自由文件）。
- ◆ Level 3(结构数据交换 , Structured Data Exchange) 人工与/或可互操作交换结构数据。
- ◆ Level 4 (无缝隙数据共享 , Seamless Sharing of Data) 在系统间基于公共交换模型自动实现数据共享。
- ◆ Level 5(无缝隙信息共享 , Seamless Sharing of Information) 通过协作数据处理实现统一的信息互操作。

最近,有人提出了新的互操作性概念模型级别(the Levels of Conceptual Interoperability Model, LCIM),同样定义了 5 个级别:

- ◆ Level 0 (系统专用数据, System Specific Data) 两个系统之间无互操作性。
- ◆ Level 1 (格式文档数据, Documented Data) 公共协议的格式文档数据和接口数据。
- ◆ Level 2 (排列静态数据, Aligned Static Data) 公共本体软件代码的公共互操作性引用模型。
- ◆ Level 3 (排列动态数据, Aligned Dynamic Data) 公共系统方法和开放源代码。
- ◆ Level 4 (协同数据, Harmonized Data) 公共概念模型, 实现语义连接。

这个互操作性概念模型体现了自下而上的接口、模型、方法到概念的相同程度的比较,值得读者注意。
这 5 级标准可以用图 16-5 表示。

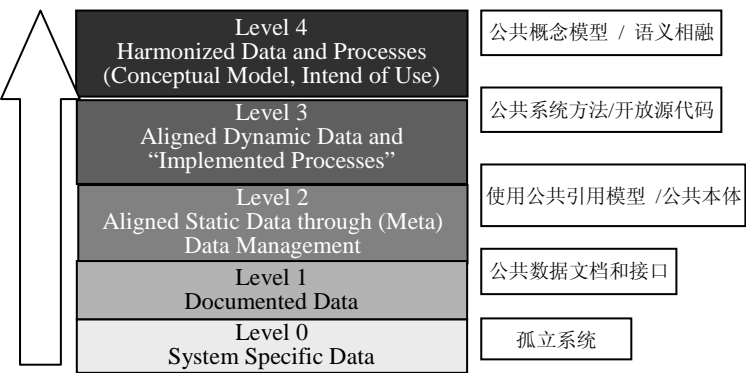


图 16-5 互操作性概念模型分级描述

第 17 章



网络世界行为学

17.1 什么是网络世界 (Cyber)

网络世界中首先有网络空间 (CyberSpace)，网络空间是一个虚拟的空间，其中有主体和客体，这些主体与客体寄宿在网络的设备与系统中。需要强调，主体是网络世界中最重要概念。其次，网络世界总体的业务概念是“运营” (CyberOperation)，其运营就是网络世界中所有主体的行为集合，如果关注点偏重信息，有时还称这种运营为信息运营 (InformationOperation)。网络世界中的运营概念是一个大概念，在这个大概念中，会产生运营互操作性、运营安全和运营服务性等重要问题。第三，网络世界中最需要关注的问题是安全 (CyberSecurity)，网络世界安全包括数据安全、网络与系统的安全、内容安全和行为安全。同时，还要为网络世界建立安全力量 (Cyber Security Force)。

有了操作系统和进程概念后，就有了计算机世界中的第一个主体，从此，在计算机中，“进程”这个主体的行为就成为了人们研究的对象。在操作系统中变化出了子进程、线程、会话、作业、任务和进程组等各有特点的新的主体形式，有空进程 (NULL)、系统进程 (SYSTEM)、用户交互进程 (JOB)、交换进程 (SWAPPER) 以及系统中断处理的辅助进程、系统管理类进程、等待进程 (daemon)、网络进程 (TCP 进程、TCP 的端口进程和 UDP 的端口进程)、应用进程和各种工具进程等。正因为如此，进程这个主体概念才成为了操作系统的核心，操作系统中的系统进程成为了操作系统行为的真正主体。在计算机和网络虚拟世界中，这个虚拟主体从诞生以来已经有近 40 年的历史了。在计算机和网络系统中，从功能上讲，这些虚拟主体主要提供系统服务，这些服务包括控制、加载、安装、初始化、配置、

生成、交换、排队、分派、调度、处理、同步、异步、维护、调试、测试、通信、交换、共享、协同、集成和管理等。随着计算机和网络应用的发展，这些虚拟主体的行为层次更加宽广，几乎涵盖了能够信息化、网络化的所有业务行为，包括银行、证券、保险、财务、税务、海关、运输、航空、商业、农业、气象、水利、电力、工业、办公、文化、娱乐、电影、电视、生活、社会、生产、安全等人类世界中几乎一切行业的业务。这种网络世界中的主体规模越来越大，虚拟主体行为的种类越来越多。更有甚者，在计算机和网络世界中，各种坏的、犯罪的、邪恶的主体以及黑客、网络恐怖主义、敌对者和战争策划者也纷纷产生，还产生了计算机病毒、特洛伊木马、蠕虫、代理软件和代理组织等各种各样的主体概念。计算机和网络的行为变得混乱和难以控制。

17.2 如何研究网络世界的行为学

由于网络世界中的主体从本质上说是软件运行的进程，所以网络世界中的行为研究也可以说是软件行为学的研究。软件行为学实际上是研究行为的语义学，这种行为的语义描述不仅是人类可读的，而且是计算机可读的。而以往的语义学是语言的语义学，并没有现成的对行为语义描述的方法。我们研究了当今国际上对行为研究的各种文献和成果，仅仅看到对单一主体的单一行为输入/输出条件满足性的行为方法。一些专家正在将人类行为的研究成果硬搬到计算机中来，这些专家被称之为人工智能学派。也可能是由于一些计算机科学家对现代信息化发展失去了了解和掌握能力，在需要多学科融合的当今时代，这些科学家提出新的理论课题的能力减弱了。也有些研究者在某些领域对代理技术与系统

的研究很多，但是对代理体系的理论研究很不深入。在这方面，《软件行为学》一书对我们是有所启发的。

为了规范网络世界中的行为，必须研究网络世界中各种虚拟主体的行为。但是，研究网络世界中不同层次的行为，包括技术与业务行为两个方面。网络世界中的行为范畴十分广阔，根据现代计算机与网络能力，到底把什么行为学的范畴作为研究对象呢？在这方面，主要出现了几种研究倾向：

1. 人工智能学派研究网络世界中的行为。

有一些科学家在研究真正的人工智能意义上的代理模型理论时，将代理作为“智能体”研究，提出了把客观的、环境的、主观的、心理的各种观念汇合在一起，构成一个所谓的模型（信念、知识、期望、目标、意图等构成的认知模型），其中，BDI（Belief-Desire-Intention，信念-期望-意图）理论模型比较出名。另外，还有一种言语行为理论（Speech Act Theory）。言语行为理论基于这样一种思想：语言表达者不仅表达了语言的意思，而且表达了一种行为。这些语言分别表示声明、委托和质询等方面的意义，所以语言学家企图借助言语行为理论表达代替意图。这种意图是从语言中反映的。语言不仅描述事物的状态，而且也描述行为，对听者施加影响或改变听者对事物的认知状况。从其基本的要素可以看出，上述行为语义的定义语言是一种纯数学的行为描述，几乎与计算机或软件没有关系。它是谓词演算的人工智能的解释，不是我们要研究的软件行为学用到的理论模型。这个理论中研究的行为是人工智能的行为，而不是计算机软件的行为。这个理论方法研究的是认知范围的问题，研究信念、目标、意图、可能性和可行性等概念，是无行为模式概念的理论体系。拿这种无行为模式的理论方法研究群体软件的组织模式、行为模

式（行为协同和行为功能模式）及行为主体的系统体系结构和技术体系结构是不行的。当前，我们感兴趣的行为模式是行为协同、行为控制、行为监管、行为认证和行为对抗；感兴趣的行为特性是行为保密性、行为完整性、行为可信性、行为有效性和行为连续性以及内容的真实性（可信性）、完整性和保密性，是对客观特性的把握，而没有主观的所谓认知、思想和心理等特性。我们研究的是克服行为方面的主观性，强化其客观性。在代理的人工智能学派中，如此的理论研究是不能指导信息化建设的。即便作为纯理论研究，似乎也仅仅是个构思。当今，我们还停留在冯·诺依曼计算机时代，虽然计算能力提高了几个数量级，但是从计算机科学来讲，丝毫没有跨越冯·诺依曼计算机模型时代。当然，从社会与生物学科中引入的这些概念对我们建立代理的体系结构仍有一定的借鉴作用。也许，在许多年后，人们需要研究软件的心理、教育学等认知模型理论。但是，我们现在需要研究的理论问题是行为控制学、行为监管学、行为保密学、行为认证学、行为对抗学和行为博弈学等范畴的较少主观意识的行为问题。笔者对于行为的本体语义、行为逻辑语义、行为态势演化语义与行为综合范畴的语义研究有自己的学术观点，如有兴趣，请参看本书的最后几章，即关于可信息化社会科学形式化、哲学形式化和信息化科学使命的讨论。

2. 数理逻辑方法学派研究网络世界中的行为。

其中，著名的语义语言（Semantic Language, SL）是 FIPA 为了定义代理行为采用的语义定义语言，主要基于谓词演算的逻辑方法，用于认知、思想和信念等范畴。采用谓词演算的方法或者数理逻辑的方法不是计算机或软件的逻辑学，而是人类的逻辑学。笔者在向学生讲授形式语义课程时曾经说过，证明与推理在

形式系统之外和证明在人类社会中是一个社会过程本身就说明，这些数理逻辑的谓词演算方法是为人建立数学形式系统的理论，是人类可读的语言，而不是计算机可读的语言。几十年的计算机科学发展史也证明：任何采用谓词演算方法来解决软件自身的逻辑问题的尝试都不能产生所希望的结果，包括程序正确性证明在内的所有这方面的理论方法都是如此，没有一个突破了人类可读逻辑的概念。除了具有说明意义外，可以说这种方法与软件实际活动（设计、开发、测试和应用等）没有直接的关系。形式逻辑学对软件和计算机发展所起的作用不够，不仅仅是从理论方法上看到的表面问题。促进事业发展的重要动力在于应用，如果始终把逻辑学作为一种科学基础来研究，而不去开拓逻辑学的应用，那么逻辑学自身的发展一定会受到限制。开拓逻辑学的应用，不能仅仅停留在应用的逻辑解释之上，而应该开拓更加贴近应用的更高层的应用逻辑体系和逻辑概念。如同程序设计不能总停留在使用机器指令或汇编语言上，而必须开拓更高层的语言体系一样，逻辑学也必须开拓新的面向应用的高层体系。其实，时态逻辑、模态逻辑、多值逻辑就是在基础逻辑理论方面的开拓，只是这种开拓距离发展的要求来说，还显得远远不够。在现实软件的开发活动中，我们建立了许多面向数据的类型库，而很少建立面向应用的逻辑类型库，就是这一问题的一个佐证。

3. 在形式语义学的诸多描述方法中，哪些适用于研究软件行为呢？

我们经历了许多形式语言，看到了许多形式说明语言的归宿。在这些理论方法中，代数理论和范畴理论方法对软件发展起到了很好的作用，类型理论对软件生产有作用，指称语义方法对语言的编译程序生成起过作用；而恰恰是逻辑语言，虽然在课堂上有

很好的描述与说明作用，也产生过 Z 语言的标准，但对软件发展的贡献不大。

4. 采用开发者视角的系统实现语言方法研究行为。

这些方法包括 FIPA 的代理通信语言 (Agent Communication Language, ACL)、REC 的 SNMP 采用的 ANS.1 方法、可扩展标记语言 (eXtensible Markup Language, XML) 方法、基于 LISP 的函数型语言 KQML (Knowledge Query and Manipulation Language, 知识查询与操纵语言) 和统一建模语言 (Unified Modeling Language, UML) 等。笔者认为这些语言方法对于研究代理行为，尤其是群体行为实在是太不够了。现在看来，它们甚至没有比 C. A. R. Hoare 在 20 世纪 80 年代初提出的通信顺序进程 (CSP) 有什么明显的进步，虽然已经过了 20 多年的时间，这些所谓的新方法依然是描述进程和并发程序设计的标准模型。由于时代发展了，在那个时期，只要解释清楚进程的静态和动态的语义概念就行了，而现在研究的问题则复杂多了，例如，提出了下面的发展要求：

1) 要全面研究应用行为模式。现在，我们不仅仅要研究进程之间的港口通信机制，还要研究代理之间在语义范畴的连接与协同，而不仅仅是通信协议概念上的连接。我们要将连接与报文的内容联系起来，还要研究代理协同起来从事的业务活动和功能活动，以及更高层次的行为模式，即我们讨论的行为类型或者模式，不能仅仅停留在计算类行为、存储行为、输入行为、输出行为、传输行为和控制行为等比较基础的行为概念上。我们研究的是管理行为、安全行为、服务行为、监管行为和对抗行为等新的更高层次的行为模式。

2) 对行为主体的研究也不能停留在并发和不确定性等概念

上。另外，将进程概念扩展到代理概念本身就带来了许多新的问题。例如，主体要构成组织，以组织的形式进行群体行为，在主体的描述中，仅仅表示并发、通信、港口和协议等概念就不够了，虽然这些概念依然是基础性的。

3) 行为的客体研究也有许多变化。在经典的并发程序设计中，我们研究的客体主要是进程间的全局变量、管道、通道、信号和事件等概念。这些概念对于高层的应用显得过于基础，距离应用层面太远。我们需要一个相互联系更为紧密的客体和对对象概念，例如，在《软件行为学》一书中提出的行为树或行为信息基概念，就将上述控制概念的客体与对象建立在一个模型之内。

4) 上述方法的一个基本不足是得不到合适的程序设计方法学的支持。下面，我们用一些篇幅来集中讨论这个问题。程序设计提出的程序结构，都由一个主程序和若干个程序模块（子程序）所组成，这表现在当今的所有程序设计语言的结构定义中，C，C++，FORTRAN 或 ADA 等语言的结构都是如此。软件的控制是在主程序或者每一个模块程序（程序包）中实现的。新型代理的程序设计概念与学科第一次提出了告别程序设计中的主程序概念，甚至非可执行代码形式的代理控制概念。利用代理中行为信息基的行为踪迹的重演，就可以实现代理的运行。怎么会这样呢？请读者注意，计算机的蓬勃发展已经有几十年的历史了，在计算机操作系统和厂商提供的系统中包含的资源如此丰富，以至于绝大多数用户不需要开发应用软件来实现他们需要的各种功能和服务，而且，几乎相同的配套系统资源分布在每一台终端计算机上。只要记录可视化运用中的操作行为踪迹，并把这些操作记录再执行，代理不就在运行了吗？事实上，现代黑客的许多攻击就是利用这种非执行代码的行为踪迹重演实现的。看到这样的现象，一个了解信息化的计算机科学家就会为此产生新的形式化模型。这

也是为什么笔者在规范代理结构中采用行为树和行为信息基（AIB）描述的原因。一个不了解现代信息化特点的人，一定会把代理的结构理解为非常复杂的传统软件系统（他忘记了他要开发的代理软件中的绝大部分功能和服务在操作系统和厂商提供的产品系统中都已经有了）。

17.3 网络世界行为语义描述方法的总要求

研究网络世界中行为的方法多种多样，但不管采用什么方法，都必须满足如下的行为语义描述的总体要求：即描述网络世界行为的本质是描述行为的语义。我们在前面说过，对于描述行为语义，许多语言是不满足要求或不充分的。那么，行为语义描述方法到底要满足什么要求呢？经过归纳，行为语义描述方法必须满足如下的总要求：

- ◆ **行为的信息综合要求** 研究代理行为的信息，不能仅仅局限于代理的输入/输出的信息内容以及输入/输出的满足性，而必须全面地提取行为的主体、客体、行为体、状态、特性、结构、模式与环境的所有可能信息。行为信息不仅是针对个体的，而且是针对群体的，包括多主体、多行为和多客体之间的关系和相关性的信息。
- ◆ **行为模式的应用要求** 行为的应用模式是指面向某种应用范畴的主体、行为、客体、环境、规则、特性和属性等方面的体系。例如，伴侣行为、行为协同、行为控制、行为监管、行为认证和行为对抗等都属于某种行为应用模式。行为模式依然是一种行为。除此之外，还应当讨论行为应用模式的本体语义。行为应用模式的本体语义

是一种应用模式自身的语义定义或描述形式。

- ◆ **行为特性的能力要求** 在网络世界中研究行为学，主要讨论的是可计算性、可理解性、可代理性、可自治性、可信性、有效性、完整性、保密性和连续性等客观特性，要研究个体和群体的行为以及行为的结果，研究内容的真实性、可信性、保密性和完整性等。为了描述行为特性、属性与规则，还应当对行为应用模式的逻辑语义、态势语义和安全语义描述方法进行讨论。
- ◆ **行为平台的系统要求** 这种语义描述方法必须在全局意义上对系统设计提供一体化的支持，不仅要支持群体中规范代理的系统结构设计，而且还要支持不同层次的群体意义上的系统结构设计，同时提供统一方式的软件工程支持。对于行为平台系统描述，我们采用行为应用模式的类型指称语义方法。

17.4 软件行为学是研究网络世界行为的语义学

笔者的《软件行为学》一书是描述网络行为的开篇之作，有很好的借鉴意义。它是基于软件行为学、开展网络虚拟世界社会学的理论与实践的研究。软件行为学是笔者在 20 世纪 90 年代中期提出的计算机科学的理论课题。《软件行为学》一书创立了行为语义学，建立了群体软件组织、模式、系统与平台结构，为信息系统代理化应用发展以及行为控制、行为监管、行为认证和行为对抗等重要领域奠定了理论基础框架。

软件行为学是抽象研究软件行为的语义学，不仅研究主体对客体的行为，还研究多主体行为之间的相互关系。软件行为学将成为计算机科学的新兴应用基础理论，是计算机与网络安全的基础理论体系；是网络虚拟世界行为监管、网络对抗、打击网络犯罪、代理技术的计算机科学的理论基础；同时还是网络网格技术的应用理论基础；既讨论软件的微观行为，又讨论软件的宏观行为；既讨论软件的个体行为，又讨论软件的群体行为。

《软件行为学》一书中建立了软件群体行为的理论框架，描述与论证了网络中群体有组织软件体系的资源使用能力、不确定计算能力、主动服务能力和自身生存能力，研究了群体中的单一代理和多代理行为，论述了类型表达式、行为树、行为信息基和多行为的行为表达式语义学方法，为形式语义学的应用发展做出了新的贡献。《软件行为学》一书中详细地阐述了行为协同、伴侣行为、行为控制、行为监管、行为认证和行为对抗 6 大行为模式；从多个应用领域讨论了多代理系统体系结构，提出了代理网格、代理网格平台、多代理、代理网络和规范代理等多层次的系统体系结构，同时还建立了一种分类型的规范代理组织结构，为代理程序设计和代理软件工程规范奠定了基础，把面向对象、面向模式、面向代理和面向行为的程序设计有机地结合起来，为程序设计理论提供了新的方法。

相对于现实社会中的行为，网络中的行为的一个非常大的不同是：在网络世界中没有非及物行为，都是及物行为。图 17-1 描述了网络世界中的各种行为形式。

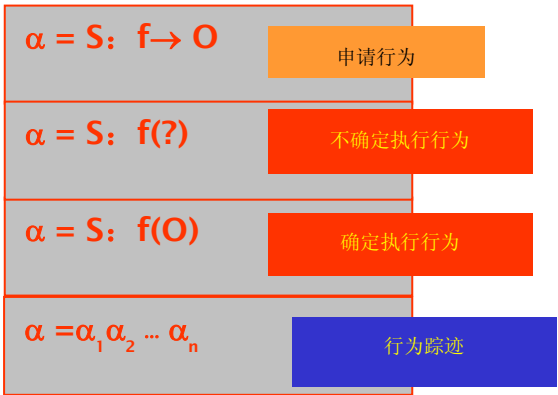


图 17-1 网络世界的行为模式

《软件行为学》一书中给出的行为表达式是描述行为模式的重要形式体系，在该行为表达式中不仅描述了单一主体（代理）的行为，而且还描述了多个主体之间的行为关系，同样描述了多代理的群体有组织行为。通过对不同主体层次行为的描述，对多主体的多行为（包括行为踪迹）给出了充分的描述能力。

Syntactic Domain:	
S : Subjects-----	主体
O : Objects-----	对象
f : Services-----	服务
F : Functors-----	函子
α, β, γ : MultiAction -----	多行为
a : AtomAction-----	原子行为
c : CompositiveAction-----	复合行为
A : ActionBody-----	行为体
Abstract Syntax :	
$\alpha ::= a$ -----	原子行为
c-----	复合行为
$\alpha; \alpha$ -----	顺序行为

$\alpha \ \alpha$ -----	平行行为(不确定时间关系的行为)
$\alpha \Rightarrow \alpha$ -----	条件单向行为
$\alpha \Leftrightarrow \alpha$ -----	对等双向行为
$\alpha' ' \alpha$ -----	选择行为
(α) -----	括弧
a ::= S : A-----	原子行为
S [O]-----	客体的伴侣行为 ,也作为原子行为
A ::= f \rightarrow O -----	申请形式的原子行为成分
f (O)-----	施用形式的原子行为成分
c ::= S ₁ (O) \Rightarrow S ₂ -----	交易类的单向行为
S ₁ (O ₁) \Leftrightarrow S ₂ (O ₂)-----	协同类的双向行为
S : F (α)-----	主体通过函子 F 对行为进行操作
S ₂ < S ₁ > : F (α_1)-----	主体 S ₂ 通过函子 F 对所伴主体 S ₁ 的行为 α_1 进行操作
S ::= ... S [O]-----	伴侣主体作为主体
O ::= ... S [O]-----	活性客体作为客体

其中，F 的默认函子名称为：SUPERVISE，CONTROL，AUTHENTICATE，DETECTION，REACTION，CONFRONTATION，ASSOCIATION 等。

根据行为的信息综合要求，研究代理行为的信息，不仅要讨论单一主体的单一行为的信息，还要综合多主体多行为的信息，这些信息包括行为的主体、客体、行为体、状态、特性、结构、模式与环境的所有可能信息。

对于我们讨论的行为而言，行为的信息概念是被“提取”(Extract) 的。例如，从行为中提取的主体、客体、伴侣、业务、

模式、状态和环境等属性及特性，就远比进程变量和港口等表达的信息丰富，行为信息提取概念远比仅考虑行为输入/输出表达的信息丰富，如图 17-2 所示。所以，如果仅仅用行为输入/输出值概念来描述行为的语义，就会限制行为研究的范畴、内容以及它们的深度与广度。

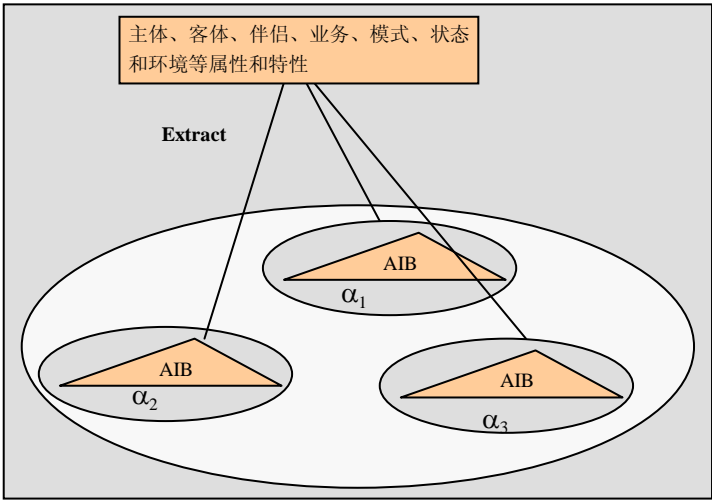


图 17-2 提取行为的信息

行为表达式虽然能反映出行为的操作，但是这种行为操作的含义却表现在行为提取的信息反映的类型值运算之上。

$A = \mathbf{Extract} \llbracket \alpha \rrbracket \rho \theta$ ，其中， α 是行为表达式反映的所有可能的行为， θ 是信息集合 A 的初始对象。 $\mathbf{Extract} : \mathbf{Action} \rightarrow \mathbf{Env} \rightarrow A \rightarrow A$ 。设用 α, β 等表示行为，用 a, a_1, a_2 等表示原子行为，用 S, S_1, S_2 等表示行为的主体，用 O 表示行为客体，用 P, Q, R, S 等字母表示断言（条件、规则等）。环境 \mathbf{Env} 是指网络世界中对于描述语义必需的定位信息的集合和用户的信息集合。

在行为逻辑表达式中有一个基本的公式：

$\alpha \text{ sat } R$ ，它是一个逻辑命题，其含义是，如果行为 α 满足断言 R ，那么 $\alpha \text{ sat } R$ 命题是成立的，否则为不成立，其含义是， $\alpha \text{ sat } R =_{\text{def}} \text{Extract} [\alpha] \rho \theta \text{ sat } R$ 。

α 是一个行为，遵守行为表达式所规定的运算，所用的符号是行为表达式中规定的符号。断言 R 是逻辑公式，这里用到的符号系统是通常的谓词演算所使用的。一个是行为 α ，一个是断言 R ，通过一个特殊的运算符号 sat 将它们联系起来， $\alpha \text{ sat } R$ 成为一个命题公式，遵从逻辑运算的规则。

$\alpha \text{ sat } R$ 的含义是什么？在行为中为什么有与断言 R 相关的特性和属性呢？我们知道行为原子成分是 $S: f \rightarrow O, S: f(O), S[O]$ 等，可以从中最终提取出主体、客体、函数、伴侣、模式和环境等信息，例如主体有标识、类型、身份、权限、应用标签（安全级别和引用范围）、状态、行为踪迹、主体规模以及应用模式关注的某些特性和属性；客体有标识、类型、拥有者标识、应用标签（安全级别、引用范围等）、对访问者要求、建立与修改日期、客体规模以及应用模式关注的特性与属性；行为中操作（服务）同样有标识、类型、应用标签（安全级别和引用范围）、对引用者要求以及应用模式关注的特性和属性；行为的模式和行为执行的环境也有其特性和属性，例如计算机的CPU序列号、操作系统序列号、BIOS序列号、应用软件序列号、网络系统序列号以及它们的名称、版本号、厂商名称、安装时间、可信测试报告等关注信息。这些特性和属性实际上是一些条件，这些条件可以与 $\alpha \text{ sat } R$ 的断言 R 进行逻辑关系的演算和推理。 $\alpha \text{ sat } R$ 是说，行为 α 提取的条件（或者说行为 α 产生的特性和属性）满足条件或断言 R 的要求（见图 17-3）。

注意，这种信息提取是在给定网络范围和多主体的多行为环

境中实施的。

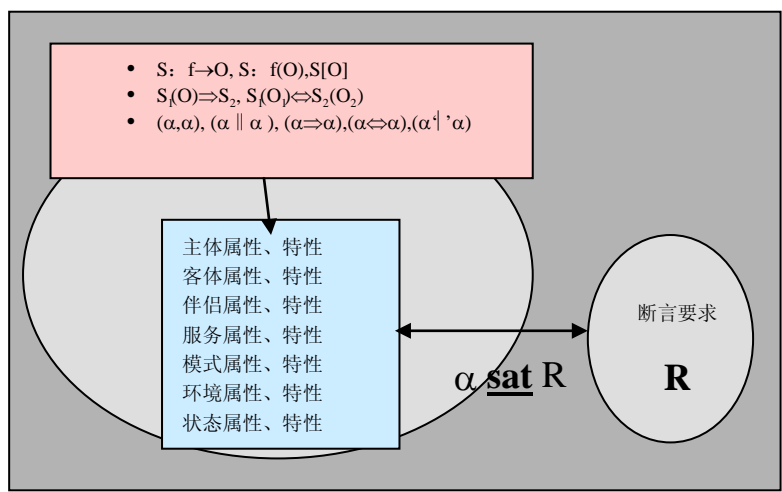


图 17-3 行为提取信息满足断言示意

需要特别说明的是，提取信息函数 **Extract** 是被关注行为的执行者之外的其他主体的行为。也就是说，**Extract** 也是一种专门的行为，同样是有主体执行的。提取信息的概念是一种关注点。显然，主体不同，关注点不同，提取信息的方法不同，提取的信息也不同。因此，存在着多主体、多关注点和多方法的问题，可以有如下 8 种类别：

- ◆ 同一主体、同一关注范围、同一信息提取方法
- ◆ 同一主体、同一关注范围、不同信息提取方法
- ◆ 同一主体、不同关注范围、同一信息提取方法
- ◆ 同一主体、不同关注范围、不同信息提取方法
- ◆ 不同多主体、同一关注范围、同一信息提取方法

- ◆ 不同多主体、同一关注范围、不同信息提取方法
- ◆ 不同多主体、不同关注范围、同一信息提取方法
- ◆ 不同多主体、不同关注范围、不同信息提取方法

为了便于讨论信息，这里主要研究单主体、同一关注范围、同一信息提取方法和多主体、同一关注范围、同一信息提取方法两个类别，其他类别我们留待以后讨论。

对于 α sat R，有如下的基本公式：

- 1. α sat TRUE，TRUE 总是真值的谓词。
- 2. $\neg(\alpha$ sat FALSE)，FALSE 总是假值的谓词。
- 3. $R \rightarrow S$ 是一个定理，行为 α 如果满足断言 R 为真，那么也必然满足断言 S 为真。有如下的公式表示：

$$\frac{R \rightarrow S}{(\alpha \text{ sat } R) \rightarrow (\alpha \text{ sat } S)}$$

$$\frac{R \equiv S}{(\alpha \text{ sat } R) \equiv (\alpha \text{ sat } S)}$$

- 4. $(\alpha \text{ sat } R) \& (\alpha \text{ sat } S) \equiv (\alpha \text{ sat } (R \& S))$ 。

在某个应用领域，所关心的信息的集合通常小于 A。例如，行为控制关注的信息集合 A_sec、行为监管关注的信息集合 A_sup、行为认证关注的信息集合 A_ca 和行为对抗关注的信息集合 A_count 是不同的。

描述行为应用模式逻辑语义，可以按照如下 4 个步骤进行：

- 1. 首先定义提取行为应用模式信息集合 A 的方法，在行为表

达式上, 针对每一个行为公式, 定义行为应用模式的信息集合的方法。通常, 由于行为表达式是递归定义的, 其描述也是递归的, 不同行为模式的关注点不同, 因此提取的信息也不同, 提取信息的函数也不同。例如, 在逻辑语义描述中, 我们定义了 **Extract** 函数, 为了简化书写, 用 E 表示 **Extract**。在 $\alpha \text{ sat } R =_{\text{def}} E \llbracket \alpha \rrbracket \rho \theta \text{ sat } R$ 这个定义中, 对函子 E 进行定义, 这种定义的方法是在对每一个行为进行逻辑定义时, 产生 $E \llbracket \alpha \rrbracket \rho \theta$ 的定义等式。将所有行为公式的中 $E \llbracket \alpha \rrbracket \rho \theta$ 的定义等式综合在一起, 就是函子 E 的最后定义。

2. 针对行为应用模式, 在行为表达式中依照 $\alpha \text{ sat } R$ 形式进行逻辑推理规则的定义。需要明确的是, 对于不同的行为应用模式, 相同的行为表达式的 $\alpha \text{ sat } R$ 形式的逻辑推理规则是可能不同的。

3. 针对行为应用模式, 利用上述在行为表达式中定义的方法, 对实际的行为模式进行代真解释, 进行实际主体行为结构的解释。

4. 对于行为命题 $\alpha \text{ sat } R$ 或 $E \llbracket \alpha \rrbracket \rho \theta \text{ sat } R$ 中的断言 R 进行规定和定义。对于断言 R , 必须针对行为应用模式中的关注点或者特性进行规定和定义。

例如, 对于行为表达式中的一些公式, 可以定义如下的行为提取信息函数公式。

◆ 行为顺序操作的提取信息函数公式:

$$E \llbracket \alpha_1; \alpha_2 \rrbracket \rho \theta = ER_{\text{SEQ}} \llbracket \alpha_1; \alpha_2 \rrbracket \rho (E \llbracket \alpha_2 \rrbracket (E \llbracket \alpha_1 \rrbracket \rho \theta)) \text{ into } A,$$

$$\text{where } ER_{\text{SEQ}}: \text{Action} \rightarrow \text{Env} \rightarrow A \rightarrow A_{\text{SEQ}}$$

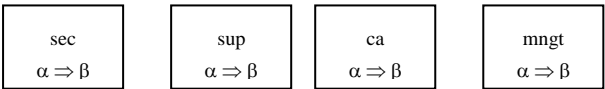
◆ 行为条件单向操作的提取信息函数公式:

$$E \llbracket \alpha_1 \Rightarrow \alpha_2 \rrbracket \rho \theta = ER_C \llbracket \alpha_1 \Rightarrow \alpha_2 \rrbracket \rho (E \llbracket \alpha_1 \rrbracket \rho \theta \parallel E \llbracket \alpha_2 \rrbracket \rho \theta) \text{ into } A \text{ where}$$

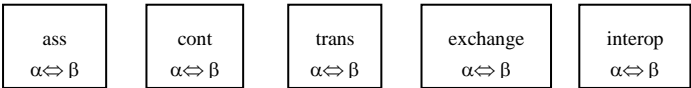
$$ER_C: Action \rightarrow Env \rightarrow A \rightarrow A_C$$

集合 A_{SEQ} , A_C 都是 A 的子集合。

根据行为学描述方法的总体要求，上述行为表达式能够描述行为应用模式。研究行为模式是把行为学理论运用到实际中的关键措施，是把形式化方法提高到应用层次的关键。例如，我们在行为表达式中定义了条件单向行为： $\alpha \Rightarrow \beta$ 。但是，我们可以用如下的单向行为关系表示方法来表示行为控制模式、行为监管模式、行为认证模式和行为管理模式：



对于双向对等行为 $\alpha \Leftrightarrow \beta$ ，可以采用如下的注释。这样，我们就可以用如下方式来表示行为协同、行为对抗、交易、交换和互操作等模式。有时，我们还可以用 $\alpha \parallel \beta$ （平行行为）来表示行为对抗模式或者同时活动的模式。



在行为表达式中，我们用 $S_2 \langle S_1 \rangle$ 和 $|S[O]$ 分别表示主体和客体的伴侣行为模式。显然，如此定义行为表达式具有应用模式的扩展能力。

需要说明的是，在描述行为应用模式时，可以采用行为应用模式本体语义的描述方法，即采用应用模式自身的语义定义或描述形式，对应用模式的主体结构、行为结构、客体结构利用行为表达式语言形式，以应用模型为依据，定义出语法、语义和语用的结构关系。

根据行为特性的能力要求,要求对个体和群体的行为可信性、有效性、完整性、保密性、连续性和行为的结果以及内容的真实性、可信性、保密性和完整性等研究具有支持能力。在每一个应用领域中,还可以细分关注的信息集合。例如,对于行为监管,我们可以关注具体行为的条件满足性的监管,也可以关注行为属性(可信性和有效性等)分类监管,还可以关注更加综合的标题性质的监管。每一种类型的监管关注的信息集合都是有差别的。在实际描述中,我们可以采用行为应用模式的逻辑语义、态势语义、安全语义等分类特性和规则方法。

行为应用模式的逻辑语义是描述行为模式的逻辑规则的语义。请读者注意,规则语义是给定网络范围内(例如全世界或国家范围内)的多主体的多行为的规则语义,包括描述行为主体、行为本身、客体所关注的规则语义,也可以是多关注的规则语义。这种规则语义描述是面向多主体、多行为和多客体的,并且研究它们的任意层次和范畴的相关性的规则。规则语义提取与给定范围的多主体多行为相关性密切相关。对相关性关注的不同,意味着提取信息范畴和级别也不同。所谓相关性,是指依据某种关系、角色、特性和模式而确立的联系。相关性分析有许多方面,例如,在语言学中,有前后文无关文法和前后文有关文法;在网络中进行相关性分析,有网络结点无关分析和网络结点相关分析。研究软件行为学,要特别研究主体的相关性和行为的相关性。

对于主体相关性,可以研究主体关系相关性和主体角色相关性等。

对于行为相关性,可以讨论行为模式的相关性和行为特性的相关性等。

在提取逻辑规则信息时,可以划分如下的一些级别:

- ◆ 单主体前后行为无关信息提取
- ◆ 单主体前后行为相关信息提取
- ◆ 多主体相关前后行为无关信息提取
- ◆ 多主体相关前后行为相关信息提取

行为应用模式的态势（situation, state）语义是行为模式的状态描述语义，行为的态势语义是行为者自身造成状态的语义描述。它不是“第三者”信息提取的语义，而是行为者主体的行为引起的状态变化的描述语义。行为的状态是指主体、伴侣主体、活性客体、客体状态与属性在行为的作用下引起的变化。

行为应用模式的安全语义是对行为的逻辑语义和态势语义求差别的语义。态势语义是描述行为实际运行时产生的体系的状态语义，是实际情况描述；逻辑规则语义是描述主体、行为和客体的规则的语义，是期望语义；安全语义描述的是在实际运行状态中，有哪些主体、行为和客体状态或属性违背了预期的规则要求。因此，这里的安全语义就是违背规则的实际行为与内容的取证，以找到一切违背规则的事件。提取证据（Extract Evidence）是安全语义的真正含义，这种证据是记录实际行为违背规则的信息。

显然，对于不同级别规则语义信息提取，其安全取证的结果是不同的。

根据行为平台的系统要求，这种语义描述方法必须在全局意义上对系统设计提供一体化的支持，不仅要支持群体中规范代理的系统结构设计，而且还要支持不同层次的群体意义上的系统结构设计，同时提供统一方式的软件工程支持。《软件行为学》一书中介绍的类型表达式、行为树和行为信息基就支持面向类型的程序设计（TOP）、面向代理的程序设计（AOP）和面向模式的程序设计（MOP）。

软件行为学的贡献就在于把数学家难于了解的各种行为应用模式展现到数学家的前面，使他们可以在行为模式平台上研究更深层次的形式化和数学理论问题。软件行为学不仅站在计算机技术的开发者视角上，而且站在 IT 技术的用户与应用视角上对网络世界中的行为进行研究。

另外，笔者与其团队编著的《银行行为监管——银行监管信息化》和《银行行为控制——银行信息化与安全》两本书，为软件行为学所提出的理论与方法进行了应用验证。

第 18 章



信息化服务理论

信息化服务理论是网络行为学理论中的一种行为模式理论。IBM 公司近些年认识到, 仅仅从 IT 厂商的角度或者计算机科学的角
度研究信息化中的问题是不够的, 因而提出了“服务科学”新概念, 期望提出新的理论、新的学说和新的知识体系, 以便从根本上改变信息化的发展状况。针对同样的现实, 我们提出的是网络或软件行为学科学方法, 而 IBM 提出的是服务科学的方法, 这是一种非常有意思的比较。

18.1 网络世界的服务行为模式

人类世界中的服务概念是指社会中的服务主体利用自己的资源、产品、业务、功能、能力、行为(行动)、特色、优势等提供服务, 使服务的受主得到满足。从软件行为学的角度看, 任何服务都是一个供应者将一种或多种行为提供给服务的接受者。

为了更好地说明问题, 给服务行为模式下一个定义:

一个主体 S_1 的行为 α_1 需要接受另一个主体 S_2 为改变 S_1 的行为 α_1 的状态、质量、能力、性质、环境、属性等而提供的行为 F , 称之为服务行为模式。其基本的表示形式有 $S_2 : F(\alpha_1)$, $S_2 < S_1 > : F(\alpha_1)$ 和 $S_2(O) \Rightarrow S_1$ 等。

我们应当了解, 如果主体 S_1 没有行为 α_1 , 那么对主体 S_1 的服务是没有意义的。不要把服务理解为直接作用在主体上的一种行为, 因为这样定义, 就把服务当成了一种操作, 而把行为接受主体看成接受操作的客体或对象。而我们在这里谈到的行为是一个主体行为的状态、质量、能力、性质、环境和属性等方面的改变。即便对于现实社会的服务概念, 如此定义也是好的。例如, 一个主体对另一个主体提供按摩服务, 对于按摩的受主来说, 实

际上是在享受、体会、感觉按摩行为引起的一种感觉。这种享受、体会和感觉如同睡觉一样是一个非及物行为。不能把这种按摩理解成按摩非生命的物体（例如一块石头），因为此时对于按摩行为的受主来说，已经不是服务了，而是一种操作。如果非要把这种操作当做服务，那么这种服务已经不是针对按摩行为的受主了，而是针对另外的主体了！为了简便，读者可以不严格地认为：服务是主体对行为的操作，是行为比较高层的概念；而基础行为是主体对客体的操作。因为这种概念定义正好来自范畴论，所以把对行为的操作定义为一种服务函子。

$S_2: F(\alpha_1)$ 和 $S_2 < S_1 >: F(\alpha_1)$ 的含义是一个主体使用行为函子 F 对行为 α_1 进行服务，也就是说，行为本身也作为操作的对象，而操作它的服务是一种称为函子或自然转换（行为或函子的射）的概念。 $S_2: F(\alpha_1)$ 和 $S_2 < S_1 >: F(\alpha_1)$ 的实质是相同的。其中，函子或自然转换 F 的默认名称有：SUPERVISE, CONTROL, AUTHENTICATE, DETECTION 和 REACTION 等，也可以统称为 SERVICES。函子或自然转换是可以由用户定义的。例如，对于 $S_2: F(\alpha_1)$ 的服务形式，可以有如下的实例：

- ◆ $S_2: SERVICES(\alpha_1)$ ，其意义是主体 S 对行为 α_1 （没有明确主体）进行服务。
- ◆ $S_2: MANAGEMENT(\alpha_1)$ ，其意义是主体 S_2 对行为 α_1 （没有明确主体）进行管理。
- ◆ $S_2: SUPERVISE(\alpha_1)$ ，其意义是主体 S_2 对行为 α_1 （没有明确主体）进行监管。
- ◆ $S_2: CONTROL(\alpha_1)$ ，其意义是主体 S_2 对行为 α_1 （没有明确主体）进行访问控制。
- ◆ $S_2: AUTHENTICATE(\alpha_1)$ ，其意义是主体 S_2 对行为 α_1 （没

有明确主体) 进行鉴别。

对于 $S_2 < S_1 > : F(\alpha)$ 的服务形式, 可以有如下的实例:

- ◆ $S_2 < S_1 > : \text{SERVICES}(\alpha_1)$, 其意义是主体 S_2 伴侣主体 S_1 , 对 S_1 的行为 α_1 提供服务。
- ◆ $S_2 < S_1 > : \text{MANAGEMENT}(\alpha_1)$, 其意义是主体 S_2 伴侣主体 S_1 , 对 S_1 的行为 α_1 进行管理。
- ◆ $S_2 < S_1 > : \text{SUPERVISE}(\alpha_1)$, 其意义是主体 S_2 伴侣主体 S_1 , 对 S_1 的行为 α_1 进行监管。
- ◆ $S_2 < S_1 > : \text{CONTROL}(\alpha_1)$, 其意义是主体 S_2 伴侣主体 S_1 , 对 S_1 的行为 α_1 进行访问控制。
- ◆ $S_2 < S_1 > : \text{AUTHENTICATE}(\alpha_1)$, 其意义是主体 S_2 伴侣主体 S_1 , 对 S_1 的行为 α_1 进行鉴别。

交易类的单向行为 $S_2(O) \Rightarrow S_1$, 实际上是 $(S_2: f(O)) \Rightarrow (S_1: f^{-1}(O))$ 复合行为的简写形式。例如 $(S_2: \text{send}(O)) \Rightarrow (S_1: \text{receive}(O))$, 其意义为主体 S_2 发送客体 O 给主体 S_1 , 主体 S_1 从主体 S_2 接收客体 O ; 其中, \Rightarrow 表示一种单向的服务行为。

对上述的服务行为进行 Lambda 抽象, 可以得到:

- ◆ $\lambda(\alpha).(S: \text{SERVICES}(\alpha))$
- ◆ $\lambda(\alpha).(S: \text{MANAGEMENT}(\alpha))$
- ◆ $\lambda(\alpha).(S: \text{SUPERVISE}(\alpha))$
- ◆ $\lambda(\alpha).(S: \text{CONTROL}(\alpha))$
- ◆ $\lambda(\alpha).(S: \text{AUTHENTICATE}(\alpha))$
- ◆ $\lambda(\alpha).(S_2 < S_1 > : \text{SERVICES}(\alpha))$
- ◆ $\lambda(\alpha).(S_2 < S_1 > : \text{MANAGEMENT}(\alpha))$
- ◆ $\lambda(\alpha).(S_2 < S_1 > : \text{SUPERVISE}(\alpha))$

- ◆ $\lambda(\alpha).(S_2 \prec S_1): \text{CONTROL}(\alpha)$
- ◆ $\lambda(\alpha).(S_2 \prec S_1): \text{AUTHENTICATE}(\alpha)$

或者对服务行为抽象，例如：

- ◆ $\lambda(\alpha).(S: F(\alpha))$
- ◆ $\lambda(F).(S: F(\alpha))$
- ◆ $\lambda(\alpha, F).(S_2 \prec S_1): F(\alpha)$

18.2 代理化的网络服务

网络世界中的服务与现实社会中的服务还是有许多区别的，首先表现在提供服务的方式上。现代信息技术服务大致可以划分为如下三种类型服务方式：

- ◆ **人员的直接服务** 人员对设备与系统的服务是不可替代的，在许多情况下，人员的直接服务是不可缺少的。
- ◆ **人员以网络远程访问方式提供的服务** 在许多人看来，只要实现网络远程的直接访问，就可以实现网络服务人员用户的直接服务，服务是技术人员高级智能的综合活动，网络服务从这个意义上讲是高科技的。我们知道，技术人员直接通过远程访问方式对计算机或网络设备提供服务，是任何机构的信息安全大忌。通过网络远程访问提供服务的通常是超级用户，这种超级用户的行为是无法被管理和控制的，更不可能对其服务进行测评认证，必须最大限度地加以避免。
- ◆ **代理与引擎软件实现的网络远程服务** 人员通过远程访问对终端计算机或设备实施服务是可以被软件所替代的。

我们所说的网络远程服务高技术化主要是指代理与引擎服务方式，这种网络服务是通过代理软件或芯片来实现的。服务代理软件是指定功能的（不多做事情，也不少做事情），是可以测评认证的，从而便于管理、控制和计费。

起源于 20 世纪 90 年代的网络管理和服务监控实现了网络代理技术的普遍应用。把管理代理软件、服务代理软件、测试代理软件与芯片、监控代理软件与芯片、维护代理软件与芯片嵌入到被管理、被服务、被测试、被监控、被维护的设备与系统中，以实现网络远程服务，是现代企业的规范和标准服务概念，能对企业降低成本发挥重要作用。但是，这也为发达国家实现网络经济意义上的控制提供了条件，直接威胁着采用此类服务的国家的信息系统、金融系统、电力系统和电信系统的安全。

第 19 章



信息化安全理论

对于中国的信息化安全事业的发展，国家、领域、企业、技术保障部门和用户的关注角度各不相同：

- ◆ **国家视角** 鉴于中国所处的国际政治、经济、外交和军事环境，国家关注主权意义上的受到侵害的网络行为、网络犯罪和网络恐怖主义行为，尤其是那些危害国家安全、社会稳定、文化侵蚀和国家基础设施等方面的行为。
- ◆ **领域视角** 领域主要关注的是领域所在方面的网络世界秩序的建立，其中包括互操作性、安全性、服务性、用户标准体系、技术法规、测评认证、信息化监管和公共资源服务等，特别关注互操作性带来的新的安全性问题。
- ◆ **企业视角** 企业对信息化的关注点通常包括信息化与企业业务与技术组织结构改革的关系，关注企业的风险监管和信息化业务运营的综合效益（包括安全效益）。
- ◆ **技术保障部门视角** 技术保障部门关注的是信息化系统的连续、安全运营，防范计算机犯罪和安全事件发生，并在发生安全事件时能够最快地恢复系统运行。
- ◆ **用户视角** 用户关注的是信息化服务功能和服务的安全，例如隐私防护、权益维护、信息安全和可信接入等问题。

我国的信息化安全建设，在前一阶段主要实现了技术保障部门关注的安全，国家、领域、企业和用户关注的安全建设在许多方面还是空白，还有巨大的市场空间。另外，信息化安全基本上是一个不断创新的行业，建立新认识、新理论和新学说对于信息化安全尤其重要。信息化安全理论是网络行为学中的行为应用模式理论。

19.1 从关注网络安全到更加关注网络运营的安全

信息化安全理论研究经历过或正在经历如下的几个发展时期。

第一时期：信息化安全的最早时期——“通信加密”时期

密码学是那时的核心安全理论，在世界范围内有近百年的历史，在中国也有 80 年的历史了，它考虑的要点是数据完整性、数据保密性和数据可用性。数据加密是主要的安全措施。

第二个时期：美国国防部的可信计算机评估准则（TCSEC）主导的安全认识时期

这个时期对信息系统安全的认识主要表现在计算机和操作系统上。TCSEC 把计算机的安全划分为 D、C1、C2、B1、B2、B3、A1 和“超 A1”8 个级别，其安全策略是对操作系统分层（将操作系统划分为核心、执行、管理与用户等层次）、分区（操作系统进程之间使用相互独立的存储空间）、分权（对不同的用户赋予不同的访问权限）、分类（依照主体与客体的安全标识或标签实现业务服务的领域隔离），其核心的安全措施是访问控制。在这一时期，已经明确了计算机系统的数据与系统的保密性、完整性和可用性等标准安全概念，依据的是 Bell LaPadula 模型、Biba 模型和基于角色的访问控制的 RBAC 等模型，采用的是有限自动机理论，企图追求安全计算机、安全操作系统、安全系统和安全网络的理想体系。从全世界来看，这种安全计算机和安全操作系统概念，除 HP 的 VVOS 应用在网关之上得到了市场的认可以外，没有得到市场的承认。在中国，由于种种原因，我们买不到高安

全级的计算机和操作系统。自主研发高安全级操作系统，又需要耗费巨资。就这样，我们用了大约 10 年的时间，才认识到必须寻求信息安全建设的新出路，找到一条适合我们自己的信息安全道路。

第三个时期：以消除计算机局域网与系统脆弱性为核心理念的发展时期

长期以来，脆弱性、通过脆弱性实现攻击、实现网络及系统内外有别的防护体系是传统安全体系的主要关注点。在相当长的一段时间里，人们比较多地关注消除系统的脆弱性和针对系统的脆弱性所产生的安全威胁。内外有别的体系结构是通过把网络与系统划分成内部（inside）和外部（outside）来防护外部威胁的。20 世纪 90 年代，随着网络应用的兴起和普及，主要的安全问题变成了网络的安全问题，这个时候的网络安全主要是从局域网或区域网角度考虑的，其安全策略是面向威胁修补系统脆弱性。安全技术方针规定为：“承认漏洞、正视威胁、适度防护、加强检测（监控）、落实反应、建立威慑”。从这些年的实践中可以看出这个方针的正确性，它得到了信息安全领域的一致支持。这个技术方针不仅适合局部环境的安全保障体系建设，也适合全局范围的安全监控及打击犯罪、对抗网络敌对势力。这个时期产生了许多技术，例如防火墙、入侵检测、漏洞扫描、审计和 VPN 等。归纳起来讲，采用的安全技术主要包括：数据安全技术，信息隐藏与发现技术，系统与网络防护技术，系统与网络安全检测监控技术，安全管理平台技术，病毒、蓄意代码检测与消除技术，身份认证技术和业务连续性技术等。

第四个时期：大范围网络环境数据与系统安全综合治理时期

在这个时期，人们认识到：安全问题不仅表现为技术、产品和系统的脆弱性，还相当程度地表现在系统和网络的结构性上，

表现在面向多系统平台、公共操作环境、公共办公环境、公共管理环境、公共服务市场等开放式的大范围环境中的互操作性和网络远程服务引发的安全问题上。互操作性带来的安全问题与网络常规安全问题和系统安全问题是不同的，这些安全问题不是单一系统的技术机理、技术体制、工程实施、运营管理体制造成的，多系统融合会造成多种技术体制、工程实施、运营管理体制的新冲突与矛盾。这种新冲突与矛盾必然会为多系统融合或互操作性带来新的安全问题。这一点对国家基础设施信息化、电子政务、电子商务和公众服务体系非常重要。这个时期的基础性信息化安全任务是把这些相互孤立的安全资源组网整合起来，构成专门的管理、监管、认证和控制的智能网络，构成有组织群体能力（高智能的分析能力、预警能力、主动服务能力和强的生存能力）的安全体系。

我们系统地提出了信息化建设的安全保障、安全监管、安全应急和安全威慑 4 大体系框架，这是信息化安全总体概念的发展与进步（参见图 19-1）。这 4 个体系不能相互替代，它们的作用是独立的。如此划分，不仅仅是由于处理这些事务的责任主体不同，更重要的是，实现的方法和技术路线也不同。

- ◆ **安全保障** 对数据与系统（计算机系统、通信系统和自动控制系统等）实施安全防护的体系，其责任主体是系统运营管理者，所以也可以称为自主安全保障体系。
- ◆ **安全监管** 对信息的内容与系统（软件/硬件）行为实施监管的体系，其责任主体是系统运营管理者 and 政府的监管当局。
- ◆ **安全应急** 确保业务连续性和防备灾害应急响应的工作体系，其责任主体是运营管理者与相应的专业组织。

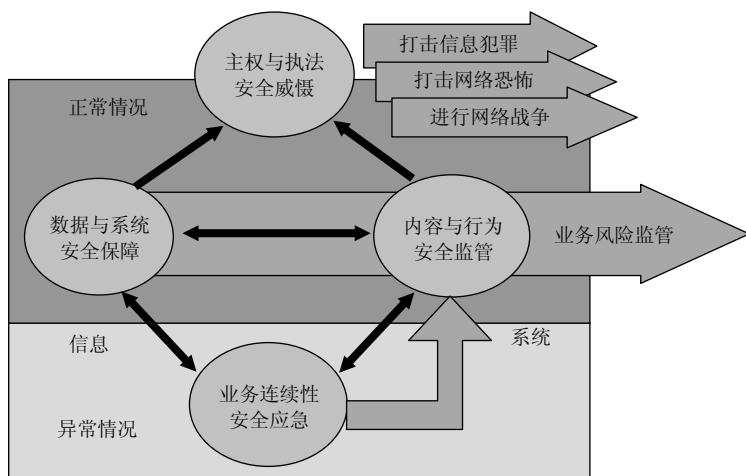


图 19-1 信息化安全总体结构

◆ **安全威慑** 打击信息犯罪、网络恐怖行为和进行网络战争的力量体系，其责任主体是政府政法与军事机构。

那种企图把信息化安全观念统一在一个非常大的“信息安全保障体系”之下的做法是错误的。

第五个时期：建立可信网络世界秩序的发展新时期

如果说在上述的四个发展时期，研究对象是数据与系统，那么在这个新时期，行为与内容是主要的研究对象，以建立可信网络世界、可信有序环境为目标，提出了从关注数据与系统安全到更加关注行为与内容的安全、从关注局域网安全到更加关注大范围网络环境的安全、从关注安全责任到更加关注安全效益、从关注面向威胁到更加关注面向能力、从关注被动安全模式到更加关注主动安全模式、从关注“授权可信”到更加关注行为可信的观念转变。这些转变是为了网络世界可信秩序的建立。这些转变首先是可信观念的转变。信息化安全的基本概念是“可信”，利用

可信与非可信的设备与系统构建信息化的安全体系，但是，传统的可信概念是通过“授权”给予的，我们称之为“授权可信”。利用这种可信概念来构建安全系统，在大多数范围内是存在严重问题的。

如何看待网络与系统的安全与可信的问题，是需要认真研究的。所谓可信，是对行为可信性预期的满足，前面已经详细讨论过了。而安全性最重要的不是考查一个系统是否符合安全技术标准、具有服务能力，而是考查这些安全功能和服务能力是否真的发挥作用，系统的安全问题是否被真正解决，是否能为用户建立安全的感知和信心，使之对实际安全效果与安全预期性之间的差距有所认识，有能力实行对这种差距的把握、控制和改变。这种安全概念实际上是通过无数的教训而获得的。考查一个网络、系统或者航天飞机的安全性，不能仅仅停留在看它采用了什么安全防护技术，而应看这些安全防护技术是否达到了安全的效果和满足了预期的要求。**可信是建立信息化安全效益的基本概念，换句话说，仅仅考虑信息化安全技术服务能力的建设，而不重视可信服务能力的建设，这种信息化安全建设就是不讲安全效益的。**

所谓可信服务，是指为建立可信能力而提供的服务。可信服务是针对网络世界主体或实体提供行为与内容的可信性、有效性、保密性、完整性和连续性等方面的服务，具体包括：行为监管服务、行为可信认证服务、行为控制服务、行为管理服务和行为对抗服务等。这个时期采用的安全技术也发生了变化。例如，可信网络世界体系结构框架（Trusted Cyber Architecture Framework，包括可信计算平台（TCP））技术、多代理技术、数字标签技术、活性认证与行为可信认证技术、监管信息化和信息化监管技术、网络对抗技术、多代理计算网格技术和系统体系结构技术等。

19.2 TCP 从不可信开启的可信网络世界 安全新模式

下面介绍计算机安全的一些问题。笔者认为，考虑可信计算平台时，必须把 TCB 与 TCP 的可信概念整合在一起；在强调 TCP 时，不能忘记 TCB，要把可信的访问控制、系统完整性、行为完整性与可信鉴别、认证和管理的平台概念结合起来：

- ◆ 基于访问控制的可信计算基概念（TCB）
- ◆ 基于系统完整性的可信计算机概念（TCP）
- ◆ 基于行为完整性的可信计算机概念（TCP）
- ◆ 建立可信鉴别、认证和管理的平台概念（TCP）

基于访问控制的可信计算基概念（TCB）

在 TCSEC 标准中定义的可信计算基(TCB, Trusted Computing Base) 概念，是一种访问控制意义上的“授权可信”。从可信的概念上看，它有其缺陷和不足，应当强调“既看身份与权限，又看行为表现”，最终落实在行为可信概念上。在 TCSEC 的可信计算机评估准则中谈到的可信概念，在访问控制范畴中是有其实际意义的。但是在 TCSEC 中规定的自主访问控制（DAC）和强制访问控制（MAC），由于 MAC 中要求的安全标识（也可以称为安全标签）的应用要求，尽管当时许多计算机跨国公司（例如 IBM, DEC, HP, Sun 和 Oracle 等）都做了 B 安全级的操作系统和相应的计算机系统，但是始终没有应用起来。因为它要求修改应用软件。但是，这种安全标识（安全标签）概念在网络中却得到了很好的应用（例如，HP 公司的 VVOS 系统开发出的网关系统就得到

了很好的应用)。在互联网中,对计算机的安全标签也有了新的认识,例如,XML 可以作为客体的数字标签使用;DAML 是美国国防部使用的代理标记语言,也可以作为数字标签使用。

值得注意的是,由于代理技术的发展,我们提出了活性标签与基于代理技术的计算机强制访问控制模式,简称活性标签与 Agent MAC 技术。它将数字标签与代理技术结合在一起,使得计算机中难于实现的 MAC 与安全标签技术可以得到实现。同时,在可信计算机的访问控制概念中,还需要引入强制行为控制(MBC)。

可信计算平台联盟(Trusted Computing Platform Alliance,简称 TCPA)是一个开发可信计算机和计算机安全技术的开放式机构。目前,TCPA 已经改组为 TCG。可信概念的定义如下:

如果一个实体的行为总是以期望的方式和意图发生的,那么这个实体是可信的。

从这个定义可以看出,可信概念建立在行为可信的基础上。可信不再是一种角色的授权可信概念,而是一种行为可信概念,是一种信誉概念。

基于系统完整性的可信计算机概念(TCP)

TCP 要求实现系统完整性,这种可信概念是通过建立几个可信的“根”来实现的,其中包括可信测量根 RTM、可信报告根 RTR 和可信存储根 RTS 等可信概念。可信测量根和可信报告根联合起来提供了平台的当前计算环境的测量数据,可以访问这些测量结果并将结果与期望的值相比较,来考查平台操作是否是所期盼的。如果测量结果与期盼值充分匹配,那么这个实体可以在平台上进行可信计算。由于这种测量和报告可以长期记录,为维护系统完整性提供了必需的信息,因此,对发现计算机病毒、特洛伊木马甚至代理入侵都有决定性的意义。这种系统完整性的测试与报告,要求开发可信平台模式(TPM)芯片。

中国信息安全产业认为，必须对可信平台模式（TPM）芯片进行代理化扩展，利用多代理技术增强可信平台测试与报告的范围和功能。

基于行为完整性的可信计算机概念（TCP）

基于行为完整性的可信计算机概念（TCP）也是 TCP 最基本的概念之一，至少是大多数国家关注的可信性问题。这个 TCP 计算模型实际上是跨国公司的方案，对于大多数国家来说，因为没有计算机核心技术和系统软件技术，所以仍然没有国家的信息安全。但是，大多数国家可以采用另一种方案，即设计可信平台的测试模块和 TPM 模块，将计算机硬件、BIOS、操作系统和应用系统等作为不可信产品实施测试，并保存测试报告，从而对系统实现监控或行为的可信性监管，对计算机厂商提供的产品是否隐藏非发布指令、非标称的存储空间和非标称的秘密接口等进行检测，而这种检测是基于行为的。只有这种安全观念才是目前的国际秩序所需要的，而不是由哪个霸权国家所控制。

同样，这种行为的完整性测试与报告要求开发可信平台模式（TPM）芯片。在行为完整性检测方面，中国信息安全产业同时还认为，必须对可信平台模式（TPM）芯片进行代理化扩展，利用多代理技术增强可信平台测试与报告的范围和功能。

建立可信鉴别、认证和管理的平台概念（TCP）

在可信计算平台概念中引入了子系统基于标识的安全鉴别、认证、加密、监管、控制和管理服务。同时，配合信息基础设施可信性与测评认证的网络服务，还要提供信元定位和测评认证的数字标签服务。

实施这项计划，既要保持与国际组织的合作与协同，又要注意维护国家的整体安全，尤其要注意，必须将可信的测试报告留在中国。

19.3 建立网络世界的独立安全理论体系

可信观念的演变经过了一段较长的认识过程，早期的可信概念反映在美国国防部的 TCSEC 标准（1985）中，其基本可信概念是“授权可信”，图 19-2 显示了其控制模式。

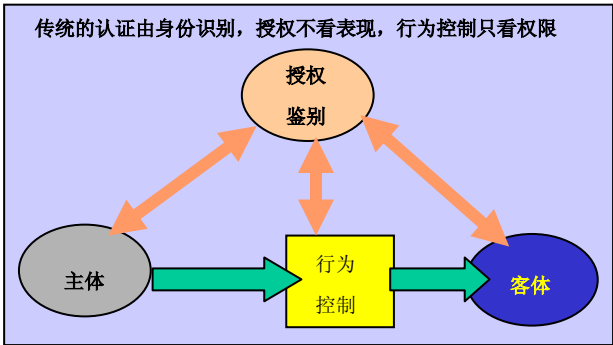


图 19-2 传统行为控制模式

在传统计算机的 TCB 强制访问控制（MAC）中，在系统主体和客体上都打上安全标签（标识），如图 19-3 所示。由于必须在每一个主体和客体上都打上安全标签，所以就要修改应用软件系统。这就使 TCSEC 的看起来非常好的安全策略失去了应用的可能性。

现代的可信概念应当是“既看身份与权限，又看行为表现”，最终落实在行为可信概念上。所谓行为可信，是指建立在对多行为或历史行为的考查（监管、认证和控制）基础之上，其行为控制模式如图 19-4 所示。下面，笔者给出《软件行为学》一书中的

描述方法，即如何实现活性标签、Agent MAC 和可信平台模式（TPM）的代理化扩展。

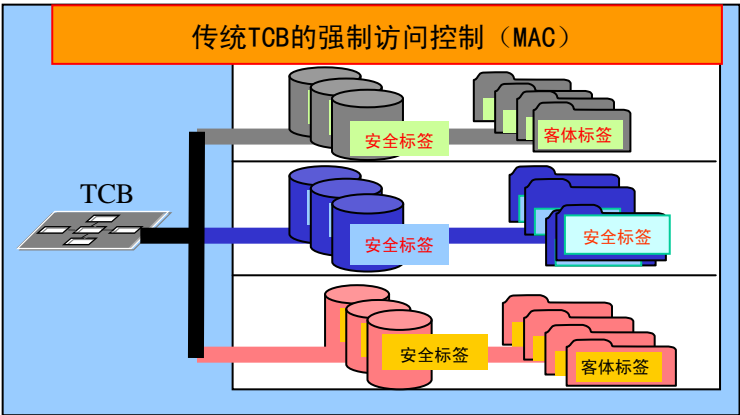


图 19-3 传统的强制访问控制模式

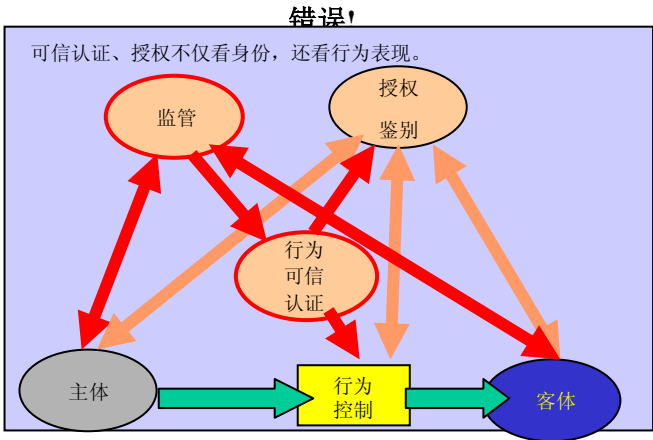


图 19-4 现代行为控制模式

实现上述的行为控制机制，仅仅依靠 TCB 的一个安全核芯片。既要检查授权、身份，又要实行行为监管，对主体的行为可信性

进行鉴别，同时对主体、客体、行为与内容进行全面的可信认证，计算机的安全核太复杂了。这种访问控制还与应用模式相关。如果实行活性标签与 Agent MAC（见图 19-5），我们同时可以解决如下方面的问题：

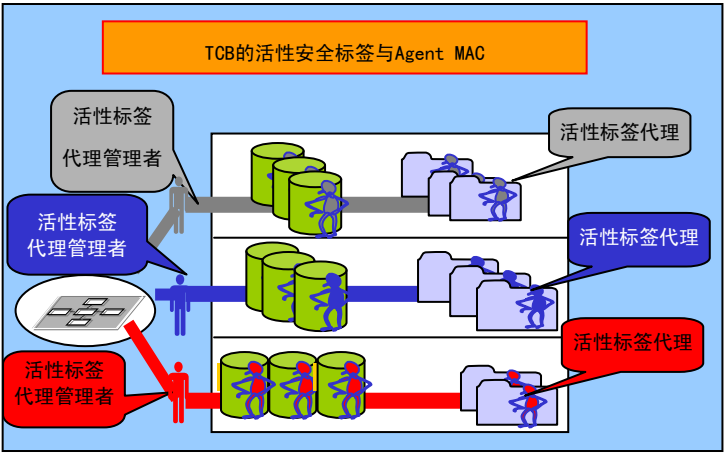


图 19-5 TCB 的活性安全标签与 Agent MAC

- ◆ 实行 Agent MAC。传统计算机的 TCB 解决的依然是主体的授权与身份鉴别。主体的可信性监管与认证，可以通过监管与认证代理去完成。
- ◆ 实行活性标签。这种标签是一个活体，由一个代理携带着，构成活性标签。这些活性标签专属于每一个主体和客体，被这些主体和客体使用。这样做的好处是不需要对应用系统进行修改，而只需在操作系统中对 TCB 实行代理化扩展。
- ◆ 上述活性标签和 Agent MAC 都发挥了多代理的有组织群体特性，其安全依靠个体与群体的两个完整性防护措施来保证，可以说是万无一失的。

总之，采用各种形式的代理技术是解决系统安全问题的重要

途径，而不管设计者或开发者给它们起什么名称。

类似地，可信平台模式（TPM）的代理化扩展可以用图19-6所示的方法来实现。

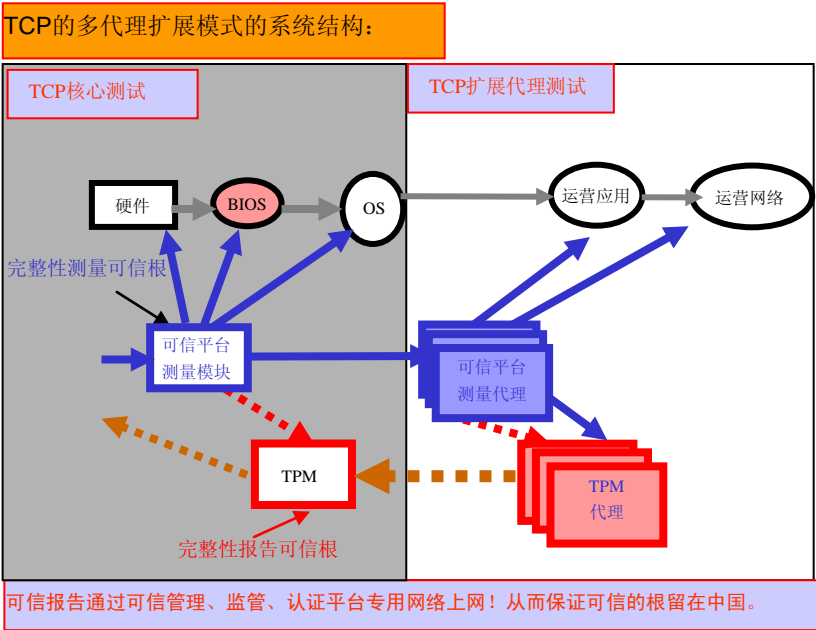


图 19-6 TCP 测试代理的扩展方案

2005 年，美国总统信息化咨询委员会（PITAC）在《Cyber Security: A Crisis of Prioritization》报告中提出了类似的结论，要求在理论和认识上进行全面的整合，包括安全理念、安全结构、身份认证、信息基础设施、软件工程、体系结构、监管与控制、减灾与恢复、取证与打击网络犯罪和相关非技术领域等。这种信息化关注点的再认识与整合，显然是由于信息化要解决的问题发生了变化，当代信息化关注的是大范围网络环境、超海量数据与对象、高智能应用与管理，显然，我们在这方面的认识与研究工

超越了 PITAC 的上述认识。

就信息化安全理论研究来说，其安全问题横跨信息化的几乎所有领域。计算机安全实际上属于计算机的理论学科，研究的是计算机软硬件及其系统的安全；通信安全实际上属于通信的理论学科，研究的是通信设备、协议、服务和网络结构的安全。这样看来，除密码学还算做安全的理论学科外，信息化的安全都依托在其他学科的理论基础上，而没有自己独立的安全理论体系。《软件行为学》一书开创的软件行为学学科是这方面的探索，期望建立信息安全自己的独立理论体系。

有了上面代理伴侣方式的安全控制技术，对于下面的行为控制形式的描述就容易理解了。在《软件行为学》一书中做了如下的行为控制模式的本体语义描述：

对于行为表达式 $\alpha_1 \Rightarrow \alpha_2$ (α_1 表示被控制行为， α_2 表示控制行为)，其行为语义为：

$$\alpha_2 = \lambda(\alpha_1).(S_2 < S_1 > : \text{CONTROL}(\alpha_1))$$

其中， $\text{CONTROL}: A \rightarrow F \rightarrow \text{ENV} \rightarrow \text{ST} \rightarrow \text{ST}$ 控制函子的类型， A 是行为信息， F 是控制方法， ENV 是控制环境， ST 是控制状态。这个表达式对控制行为进行了定义， $S_2 < S_1 >$ 表示代理 S_2 是主体 S_1 的伴侣，控制代理 S_2 对主体 S_1 的行为 α_1 实行控制。控制代理 S_2 可以依据安全标签实行控制。

下面，我们举一个例子来说明这种形式化方法的意义：

$$\alpha_0 \Rightarrow \beta_0$$

令 $S = \{S_0, S_1, S_2, \dots, S_n\}$ 表示一个主体集合，其组织关系可以表示为 $S_0 :: S_1 S_2 \dots S_n$ ， S_0 是管理者； $A = \{A_0, A_1, A_2, \dots, A_n\}$ 表示一个安全控制代理的集合，其组织关系可以表示为 $A_0 ::$

$A_1 A_2 \cdots A_n$, A_0 是管理者。其中, 主体 S_1, S_2, \cdots, S_n 与安全控制代理 A_1, A_2, \cdots, A_n 构成一一对应的伴侣形式, 即: $A_1 < S_1 >$, $A_2 < S_2 >$, \cdots , $A_n < S_n >$ 。安全控制代理同时也是主体的安全标签代理。

令 $O = \{O_1, O_2, \cdots, O_m\}$ 表示客体的集合。

令 $\text{ActiveLabel} = \{\text{ActiveLabel}_0, \text{ActiveLabel}_1, \text{ActiveLabel}_2, \cdots, \text{ActiveLabel}_m\}$ 表示另一个活性标签代理的集合, 其组织关系可以表示为 $\text{ActiveLabel}_0 :: \text{ActiveLabel}_1 \text{ActiveLabel}_2 \cdots \text{ActiveLabel}_m$ 。其中, $\text{ActiveLabel}_1, \text{ActiveLabel}_2, \cdots, \text{ActiveLabel}_m$ 与客体 O_1, O_2, \cdots, O_m 构成一一对应的伴侣形式, 即: $\text{ActiveLabel}_1[O_1], \text{ActiveLabel}_2[O_2], \cdots, \text{ActiveLabel}_m[O_m]$, 因此有 $\text{ActiveLabel}_0 :: \text{ActiveLabel}_1[O_1], \text{ActiveLabel}_2[O_2], \cdots, \text{ActiveLabel}_m[O_m]$ 。

上述的主体安全控制代理中的安全标签与活性客体的安全标签代理依照计算机的强制访问控制 (MAC) 机制, 具有对应关系。

令 γ_i 是主体 S_i 的组织行为, $i \in \{1, 2, \cdots, n\}$, 并有 $\gamma_0 = S_0$: $\text{Management}(\gamma_1 \parallel \gamma_2 \parallel \cdots \parallel \gamma_n)$ 。

令安全控制代理 (主体安全标签代理) 组织的行为用 χ_i 表示, $i \in \{1, 2, \cdots, n\}$, $\chi_0 = A_0$: $\text{Management}(\chi_1 \parallel \chi_2 \parallel \cdots \parallel \chi_n)$ 。

下面对主体 S_i 的业务安全控制行为 ($\alpha_0 \Rightarrow \beta_0$) 进行描述, 令 α_i 是主体 S_i 的业务行为, 定义为 $\alpha_i = S_i : f \rightarrow O_j$, $i \in \{1, 2, \cdots, n\}$, $j \in \{1, 2, \cdots, m\}$ 。其中, f 是一个函数或操作, 并有 $\alpha_0 = S_0$: $\text{Management}(\alpha_1 \parallel \alpha_2 \parallel \cdots \parallel \alpha_n)$ 。

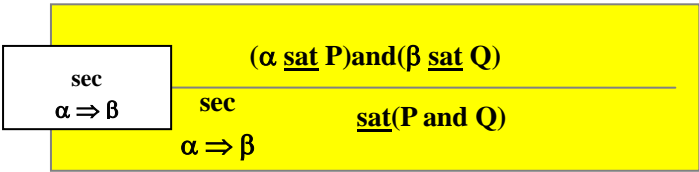
令 β_i 是安全控制代理 A_i 的安全控制行为, 定义为 $\beta_i = \lambda(\alpha).(A_i < S_i > : \text{CONTROL}(\alpha))$, $i \in \{1, 2, \cdots, n\}$, 并有 $\beta_0 = A_0$: $\text{Management}(\beta_1 \parallel \beta_2 \parallel \cdots \parallel \beta_n)$ 。

$\alpha_0 \Rightarrow \beta_0$ 表示安全控制代理对系统主体的行为 α_0 实行控制行为 β_0 , 并且有如下关系:

$$\begin{aligned}\alpha_0 \Rightarrow \beta_0 &= A_0 \langle S_0 \rangle : \text{CONTROL}(\alpha_0) \\ &= (\alpha_1 \Rightarrow \beta_1 \parallel \alpha_2 \Rightarrow \beta_1 \parallel \cdots \parallel \alpha_n \Rightarrow \beta_1) \\ &= (A_1 \langle S_1 \rangle : \text{CONTROL}(\alpha_1) \\ &\quad \parallel A_2 \langle S_2 \rangle : \text{CONTROL}(\alpha_2) \\ &\quad \parallel \cdots \cdots \\ &\quad \parallel A_n \langle S_n \rangle : \text{CONTROL}(\alpha_n))\end{aligned}$$

其中，控制函子 CONTROL 要根据安全控制代理 $A_i(i \in \{1, 2, \cdots, n\})$ 以及活性标签 $\text{ActiveLabel}_j(j \in \{1, 2, \cdots, m\})$ 进行主体 $S_i(i \in \{1, 2, \cdots, n\})$ 对客体 $O_j(j \in \{1, 2, \cdots, m\})$ 符合 MAC 规则的访问控制。

假定 α 是被控制行为， β 是控制行为，那么行为控制逻辑语义描述如下： $\alpha \Rightarrow \beta$ 。如果利用逻辑思考的原理，显然只有在被控制行为 α 是可信的（满足条件 P），同时 β 是可信和有效的（满足条件 Q）的情况下， $\alpha \Rightarrow \beta$ 才是可信和有效的（满足条件 P and Q），控制可以通过。如果被控制行为是不可信的（不满足条件 P），或控制方法（例如访问控制方法）是不真实的（不满足条件 Q），那么控制是通不过的（不满足条件 P and Q），用公式表示如下：



第 20 章



信息化监管理论

20.1 从监管业务进网谈起

信息化监管理论是网络行为学中的一种行为应用模式理论。这里谈到的许多研究成果都是在笔者与朋友合著的《银行行为监管——银行监管信息化》一书中首次提出的。

传统的评估或者监管工作主要是事后检查和审计，这项工作主要通过查账进行，是靠人工完成的。在银行实现信息化、网络化，以“光电的速度”处理事务的年代，服务实现了网络化，可以说，“服务在网络里”。金融监管与审计不能只靠“人的眼睛”，监管是人工的，靠看打印结果或屏幕查询实施监管，可以说，“监管在网络外面”。这种“服务在网络里”和“监管在网络外面”之间的速度不匹配，造成了评估或监管的虚假性。必须研究在信息化条件下的新的监管理论和实践。

人类世界和网络世界的风险是不同的，监管也是有显著差别的。人类世界的风险监管主要是面对人类的行为及其行为结果进行的，其监管是对人类的行为和行为结果不确定性、危险性、灾害性、损失性、安全性和生存性进行关注的系统过程。**这个监管过程虽然可以得到信息化的帮助，但是活动和监管过程的速度是相匹配的，活动在网络外面，监管也在网络外面。**网络世界的风险监管是面对网络世界中主体的行为和行为结果进行的，其监管是对网络世界中主体行为与行为结果不确定性、危险性、灾害性、损失性、安全性和生存性进行关注的系统过程。在网络世界中实施监管不能依靠人对管理信息系统监视，只能依赖监管多代理系统实现“服务在网络里”和“监管也在网络里”的基本要求。

人类世界和网络世界的风险与相应的监管是相关联的。不仅要注意到这两个世界风险监管的区别，同时还要注意把它们结合

起来。现代银行业务与技术活动，是人类行为以及人类使用信息化的行为和网络世界中虚拟主体行为的组合体系。因此，银行信息资产风险监管也必须把人类世界银行业务活动的风险监管与网络世界的风险监管构成一体化的体系。这种一体化体系是通过人类活动的信息化管理系统和网络世界中虚拟主体活动的管理系统结合实现的。

20.2 要建立网络世界风险监管的多视角认识体系

监管概念在实践中首先应用于风险监管。对于风险监管，有多种研究方法，例如对于金融风险监管，就有如下几种方法：

风险监管的金融学方法

风险监管的金融学方法主要用于为风险评分与定价，为管理、监管、信息化进行评估与定价，其核心理念是将风险价值化。从金融的观点来看，以资金、资产、价值的观念研究风险监管，要把管理、监管、风险、损失等一切概念都与“钱”或者价值联系起来，把风险价值化之后，在银行统一的价值计算体系内对风险进行评估和管理。风险监管的金融方法学中的典型成果是巴塞尔资本协议，它指明了风险监管的价值化准则，提出了三大风险监管支柱。第一支柱：最低资本要求。1. 信用风险：债务人或交易对手违约造成损失的风险（违约概率、违约损失率）。2. 市场风险：价格的不利变化使交易头寸蒙受损失的风险。3. 操作风险：由于内部程序、人员和系统的不完善和失误或者外部事件造成的直接或间接损失的风险。第二支柱：监管部门鼓励银行采用

更好的风险管理技术来监测它们的风险。第三支柱：市场纪律，主要是对监管实施信息披露，加强市场的纪律作用。

风险监管的系统工程方法学

风险监管的系统工程方法学主要是风险因素对策方法学，是从系统工程的角度，将风险监管纳入统一的因素分析体系中，来研究减少或规避风险的方法。从某种意义上说，它是在微观的概念范围内研究风险监管的。风险监管的系统工程方法包括系统风险概念、系统风险特性、价值分析、攻击威胁分析、脆弱性分析、健壮性模型与分析、信息安全模型与分析、可靠性与业务连续性分析、可管理性与可监控性分析、生存性分析等。

风险监管的管理学方法

风险监管的管理学方法以社会行为学或组织行为学为基础，全面系统地研究人类世界、物理世界中的活动主体对风险监管的意义。社会科学的组织行为学与计算机科学的软件行为学相互对应，相互支持，共同构建了风险监管的理论体系。从法律、制度、工作程序、责任、管理体制、人员、风险评分指标体系、定价体系以及管理信息化等方面，将风险监管责任化，将监管工作纳入统一的责任体系内实现评估与管理。

风险监管的信息化科学方法

风险监管的信息化科学方法是我们提出的网络世界的行为学方法。该方法提出，风险监管主要研究行为的可信性、有效性、完整性和保密性，以及内容的完整性、保密性与可信性（或真实性）。银行业务的网络化、信息化和传统的风险监管模式之间存在严峻的矛盾。我们必须全面改变银行“服务在网络内，而监管在网络外”的尴尬局面，银行风险监管信息化的根本发展途径是银行业务与技术的监管进网。这方面的内容在《软件行为学》、

《银行行为监管——银行监管信息化》和《银行行为控制——银行信息化与安全》等专著中有详细的介绍。研究软件行为概念、行为特性、行为状态、行为生存期、行为协同、伴侣行为、行为控制、行为监管、行为认证、行为对抗和行为平台等是风险监管信息化和信息化监管的基础理论。对于已经实现了业务信息化和网络化的金融领域，如果简单地将物理世界的风险监管模式克隆到网络世界中，本身就是金融行业最大的风险。

从金融学家的角度看风险监管与从系统工程的角度看风险监管和从管理学的角度看风险监管是很不相同的，必须把这些从不同角度考查风险监管的方法结合起来，建立一个比较科学的决策、协调体制，才能实现风险监管的综合治理。任何个人的知识结构与工作经历都是有局限性的，单一角色的决策体制是最危险的风险监管体制。例如，经济学家和金融学家决策工业类的、工程类的问题，往往会产生一些错误的结论。金融学家看风险，最后会把风险都用“钱”来衡量，把风险资产化后，有时会忽略风险的真实因素。同样，仅仅从系统工程的方法来研究风险，会抓住风险形成的因素，关注事务成败的本身，而很少去研究风险的价值。如果让系统工程师来决策经济与金融类市场问题，也会造成抓住微观而忽略宏观的另一类错误。管理学家研究风险，考虑的是如何建立一种体制、制度来管理和控制风险，他们应该了解风险管理概念和系统工程的风险因素的分析方法，来弥补管理学的不足。

银行家、经济学家、系统工程师、管理学家采用的风险监管方法是很不相同的。我们总说，实现风险监管需要综合治理，但这种综合治理不是有了主观愿望就可以实现的，必须把他们的风险监管方法结合起来，让每一个人了解不同风险监管方法只是综合风险监管的一个组成部分，还应该了解每一种方法在综合监管中的位置角色及其所能承担的任务。首先，应达成认识上的一致，

在理论、方法、体系和原则的框架上建立统一的认识体系平台。在这个认识体系平台上，银行家、经济学家、管理学家、系统工程师再实践各自新的风险监管方法，去追求一个新的高度。银行风险监管涉及到金融学、管理学、经济学、系统工程、信息化科学，而不能仅仅用金融学方法来单纯地研究银行风险监管。

20.3 用信息资产风险监管来构建监管体系核心理念

监管现代化体系必须建立一体化的认识平台，这个认识平台就是信息资产风险监管体系。落实监管业务进网，要以监管原始数据和原始行为为基础，实现风险的分类与全面的综合监管，建立信息资产风险评分与定价体系和评级标准。

1. 什么是信息资产

信息资产 = 信息范畴资产 + 系统范畴资产 + 附加范畴资产。其中，信息范畴资产是指信息存在形式、信息内容和内容价值；系统范畴资产是指系统存在形式、系统行为和行为价值；附加范畴资产是指不同的关注者（计划者、拥有者、设计者、实施者和用户等）关注的信息与系统两个范畴的需求价值（附加价值），例如开发、运营、管理、维护等产生的关注性的资产。

对于信息而言，信息存在形式是指信息存在的成本价值，信息内容主要指信息的类型和所能提供的信息内容范围，内容价值是指对内容被引用和内容本身产生的效益情况的估算。系统范畴资产中的系统存在形式是指系统建设和存在的成本风险，系统行为是指系统所能提供的服务，而行为价值是指系统服务被引用和服务产生的

效益估算。要注意信息系统不能提供内容与服务时所产生的损失价值。这种价值有时按照分、小时、天、星期、月等来计算。需求价值表示的是一种市场供求关系价值，是一种附加值。

2. 什么是信息资产风险

信息资产风险是指在信息资产的规划、设计、开发、生成、存在、运用、服务、管理、维护、监管以及其他相关过程中产生的信用、市场、操作与业务风险。其中，信用风险是指交易违约行为及其损失的统计风险；市场风险是指价格引起的交易损失的统计风险；操作风险是指内外人员、系统在运营和管理中的行为造成的直接或间接损失风险；业务风险是指业务所派生的风险。

3. 信息资产风险监管

面对风险进行的信息采集、测试、分析、评估、预案、设计、接受、管理、控制、优化、规避、化解、应急等方面的过程，称之为风险监管。

20.4 网络世界风险监管理论体系

从网络世界行为学理论来看，应当明确如下的一些基本概念。

1) 监管的对象

银行监管信息化体系需要监管人类世界中人类的相关行为和
网络虚拟世界中虚拟主体（代理）的相关行为与行为结果（内容）。

图 20-1 描述了安全监管与安全保障之间的分类关系。

2) 信息资产风险行为监管的任务

我国信息资产风险行为监管的主要任务包括风险监管信息化
和信息化风险监管两个方面。风险行为监管包括行为条件基础监

管、行为属性分类监管和行为标题综合监管。

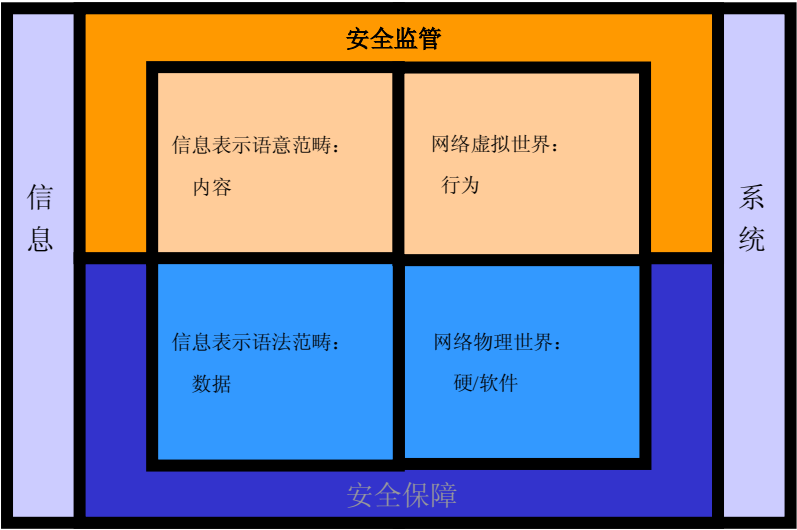


图 20-1 安全保障与安全监管

行为条件基础监管主要是对行为的自然属性和行为条件进行检查，例如主体、行为名称、行为对象、行为类型、行为输入和行为输出等。这种监管还包括行为输入、输出、操作、执行要求的条件满足性检查。

行为属性分类监管是在行为条件基础监管的基础上，对行为的有效性、可信性、保密性、完整性和连续性，内容的真实性、可信性和保密性等特性进行分类监管。

行为标题综合监管是在行为条件基础监管和行为属性分类监管的基础上，实行更高层次的综合监管，这种综合监管通常用一个监管的服务标题加以命名。例如：

◆ 行为综合监管

- ◆ 信息汇总监管
- ◆ 政策传导（畅通性）监管
- ◆ 流量流向监管
- ◆ 风险（信用风险、市场风险和操作风险等）监管

3) 风险评估与定价体系

业务与管理信息化以及风险监管信息化，不仅要求建立一些管理制度，还要求建立相当充实的风险监管和评估指标体系。建立风险评估指标体系是做好风险监管的根本。建立风险评估指标体系，应考虑以下几个方向：

- ◆ 风险处理态度划分级别
- ◆ 风险监管评级指标
- ◆ 风险监管工作评估指标体系
- ◆ 风险监管人员评级指标体系
- ◆ 风险监管人员绩效考评指标体系
- ◆ 风险监管成熟度量指标体系

4) 风险处理级别

人们对待风险通常有接受、应急管理（应急计划、应急管理、应急控制）、风险管理（风险计划、风险管理、风险控制）、过程控制（全局计划、全局管理、全局控制）等不同层次的处理态度。在处理风险时，始终要考虑风险规避与化解。

5) 风险监管评级指标

《银行行为监管——银行监管信息化》一书中推荐的风险监管级别依次为：

- ◆ 方法监管（员工与部门视角） 所谓方法监管，是指采用金融学、经济学和管理学的方法，在业务范围内针对某些

风险或危害对项目或企业承担风险的能力进行风险监管评价、评级和评分，从而采取相应的措施规避、化解、优化、计划、控制和管理这些风险。

- ◆ **结构方法监管（企业领导视角）** 结构方法监管是在认识到产生风险的深层次、结构性和全局性的原因之后，采取更加深刻与全面的管理措施。它更多地研究各种风险之间的相互关系，强调风险监管结构布局、层次布局，期望逐步堵塞监管的漏洞。
- ◆ **行为监管（股东和董事会视角）** 在结构方法监管的基础之上，全面解决价值范围内监管的有效性、可信性、连续性、真实性，实施对业务行为和技术行为的监管（包括内容监管），确保监管行为与监管数据的可信与真实，同时强调防止监管行为走过场，强化监管行为的有效性和连续性。
- ◆ **结构行为监管（领域和监管当局视角）** 在上一级别的基础上，实行单一企业或多企业在领域范围内的结构性行为监管，行为监管的漏洞得到全面弥补。
- ◆ **多结构行为监管（多国家监管当局视角）** 在国际范围内实行企业的结构行为监管，涉及到多个国家的监管当局的监管问题。

在讨论监管级别时，我们把方法监管放在最低的级别上，这种级别安排是先易后难，最后还得通过行为监管、结构行为监管和多结构行为监管来实现真实、可信和有效的监管。

从后面将要谈到的体系结构概念来看，这样的划分是具有理论依据的。正如图 20-2 所表示的那样，方法监管主要是从事业务的企业员工关注的事情，其方法是面向业务的，其监管对象主要是业务的对象。结构方法监管是企业领导层关注的概念，其性质

是管理范围，其主要任务是综合与协调整个企业的各项业务，采用分类与综合方法进行监管。当然，这种结构性的关注点是建立在企业所有员工方法监管的关注点基础上的。行为监管是企业拥有者或者股东关注的概念，监管的性质是价值范围，以求得企业风险的最小化。可以看出，此时监管对象已经发生了变化，即企业的员工和企业的领导层也纳入了监管范围。当然，企业员工与客户的行为，内容的真实性、可信性和有效性，与企业投资价值密切相关的属性是监管的主要对象。结构行为监管把这种价值概念的监管扩大到企业的全局和领域范围之内，以求得整个企业和领域的有序发展、价值的最大化以及风险的最小化。多结构行为监管把结构行为监管扩大到国家和国际范围内。

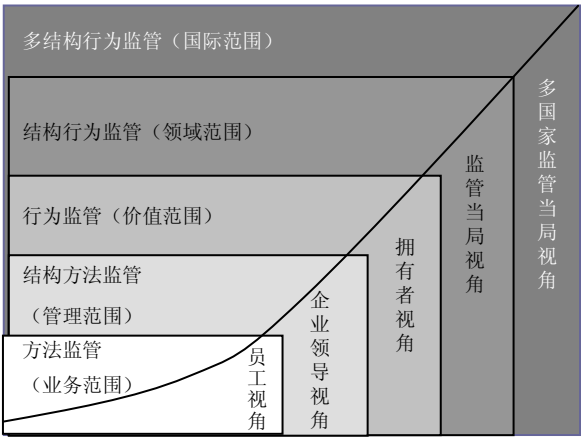


图 20-2 监管等级标准示意

显然，如果需要，可以进一步细分上述的监管级别。

6) 行为监管的技术路线

对主体行为进行监管，要设计一个监管的代理，与主体捆绑在一起，对主体的每一个行为都实行监管。这种捆绑方式就是我

们谈到的代理伴侣主体的一种行为模式，如图 20-3 所示。

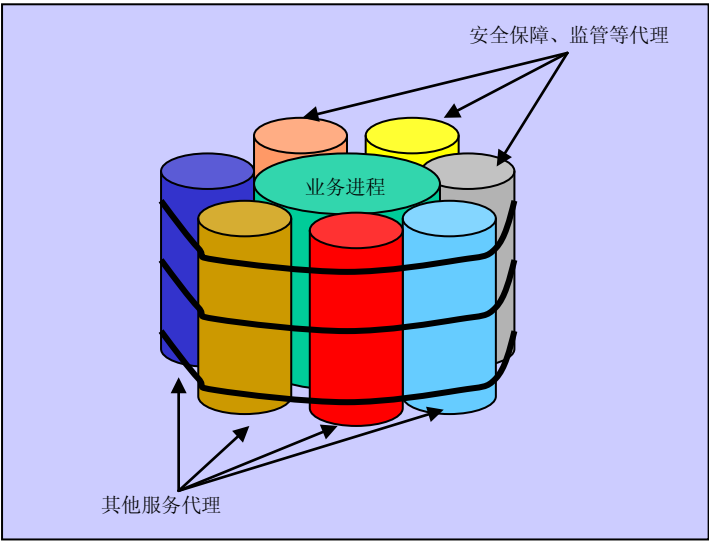


图 20-3 监管代理与业务进程捆绑

具体的实现方法是为监管代理设计一个行为树，在树上遍历时，可以把行为树划分成两个部分（见图 20-4）：预期行为和非预期行为。预期行为部分又分为业务行为和办公行为，这些行为是加分行为。用业务行为和办公行为的时间与总办公时间的相对比值表示加分的数值。非预期行为是减分的，分为一般性非预期行为、具有危害性的非预期行为和具有高危害性的非预期行为。不同级别的非预期行为的减分数值不同；对于有些行为，不仅减分，而且还要发出警告或进行处罚。

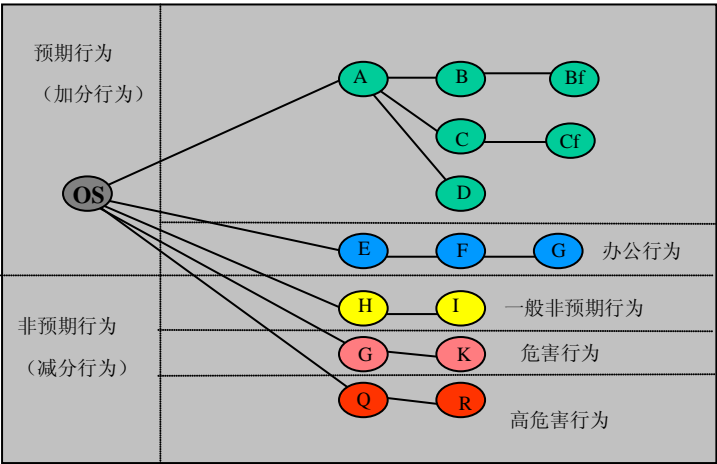


图 20-4 监管代理行为树分级

20.5 行为监管的形式化描述

行为监管是整个监管可信与有效的重要保障，是行为条件基础监管和行为属性分类监管的主要措施。《软件行为学》一书中做了如下的行为监管本体语义描述：

对于行为表达式 $\alpha_1 \Rightarrow \alpha_2$ （其中， α_1 表示被监管行为，而 α_2 表示监管行为），其行为语义为：

$$\alpha_2 = \lambda(\alpha) . (S_2 < S_1 > : \text{SUPERVISE}(\alpha))$$

其中， $\text{SUPERVISE} : A \rightarrow F \rightarrow \text{ENV} \rightarrow \text{ST} \rightarrow \text{ST}$ 监管函子的类型，A 为行为信息，F 为监管方法，ENV 为监管环境，ST 为监管状态。下面举一个例子来说明这种形式化方法的意义。

令 $S = \{S_0, S_1, S_2, \dots, S_n\}$ 表示网络中的一个业务主体集合，其组织关系可以表示为 $S_0 :: S_1 S_2 \dots S_n$ ；其中， S_0 是管理者，而 S_1 ,

S_2, \dots, S_n 是被管理的主体。 $A = \{A_0, A_1, A_2, \dots, A_n\}$ 表示一个监管代理的集合, 其组织关系可以表示为 $A_0 :: A_1 A_2 \dots A_n$, 是 $S_0, S_1, S_2, \dots, S_n$ 的监管代理的组织形式, 与 $S_0 :: S_1 S_2 \dots S_n$ 结构相同, 一一对应, 即 $A_1 < S_1 >, A_2 < S_2 >, \dots, A_n < S_n >$ 。

令 γ_i 是主体 S_i 的组织行为, $i \in \{1, 2, \dots, n\}$, 并有 $\gamma_0 = S_0$:
 $\text{Management}(\gamma_1 \parallel \gamma_2 \parallel \dots \parallel \gamma_n)$;

令 χ_i 是监管代理 A_i 的行为, $i \in \{1, 2, \dots, n\}$, 并有 $\chi_0 = A_0$:
 $\text{Management}(\chi_1 \parallel \chi_2 \parallel \dots \parallel \chi_n)$

令 α_i 是主体 S_i 的业务行为, $i \in \{1, 2, \dots, n\}$, 并有 $\alpha_0 = S_0$:
 $\text{Management}(\alpha_1 \parallel \alpha_2 \parallel \dots \parallel \alpha_n)$;

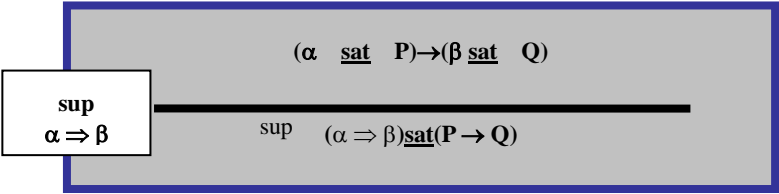
令 β_i 是监管代理 A_i 的监管行为, $\beta_i = \lambda(\alpha_i). (A_i < S_i > : \text{SUPERVISE}(\alpha_i))$, $i \in \{1, 2, \dots, n\}$, 并有 $\beta_0 = A_0 : \text{Management}(\beta_1 \parallel \beta_2 \parallel \dots \parallel \beta_n)$;

$\alpha_0 \Rightarrow \beta_0$ 表示监管代理对业务主体的行为 α_0 实行监管行为 β_0 , 并且有如下关系:

$$\begin{aligned} \alpha_0 \Rightarrow \beta_0 &= A_0 < S_0 > : \text{SUPERVISE}(\alpha_0) \\ &= (\alpha_1 \Rightarrow \beta_1 \parallel \alpha_2 \Rightarrow \beta_2 \parallel \dots \parallel \alpha_n \Rightarrow \beta_n) \\ &= (A_1 < S_1 > : \text{SUPERVISE}(\alpha_1) \\ &\quad \parallel A_2 < S_2 > : \text{SUPERVISE}(\alpha_2) \\ &\quad \parallel \dots \parallel \\ &\quad \parallel A_n < S_n > : \text{SUPERVISE}(\alpha_n)) \end{aligned}$$

行为监管还可以采用逻辑语义描述方法。对于 $\alpha \Rightarrow \beta$, 如果利用逻辑思考的原理, 那么只有在被监管行为 α 可信 (满足条件 P) 和 β 可信 (满足条件 Q) 的情况下, $\alpha \Rightarrow \beta$ 才是可信和有效的 (满足条件 $P \rightarrow Q$)。如果被监管行为是不可信的 (不满足条件 P), 那么即使监管方法 (金融学方法) 是真实的, 监管也依然是虚假的 (欺诈、假票据、假账等), 因此, $\alpha \Rightarrow \beta$ 应当满足的条件是 P

→ Q，如下所示：



从上面的讨论中可以看出，监管不是控制，监管不干预被监管行为的过程。因此，监管在逻辑上比控制要弱。

第 21 章



信息化认证理论

信息化认证理论是网络行为学中的一种行为应用模式理论。信息化的发展正从局部、部门和企业网络范围的信息化建设向大范围网络环境、超海量数据对象与高智能应用和管理转变；从“宽带高速”的发展战略向“有效利用资源”的发展战略转变；从面向个体应用模式向面向代理化群体应用模式转变；在计算模型方面，更加关注群体计算理论与实践。整个 IT 行业酝酿着重大的变革。在群体计算模型、体系结构以及多代理技术研究与应用方面的成果，为未来的信息化带来了新的希望。在这样的形势下，认证理论与技术研究也在寻求全面的突破和发展。

21.1 公众服务认证需要组合密钥（CPK）

在认证理论与技术的研究中，基于 PKI 技术实现的认证系统是当代最著名的体系，但是，PKI 由于其第三方在线认证技术体制，造成了认证规模小、资源需求大、管理与运营成本高和机构复杂等问题。面对电子政务、电子商务和基础设施信息化的大规模认证（例如数十亿、数百亿规模的认证单位），这种认证技术体制是不可能适应的。另外，该方法不适合网络世界中代理、进程和主体的身份认证。其二，还有一种 IBE（Identity Based Encryption）算法。它虽然是第三方认证技术，但依然保留了在线认证服务，需要保留大量的用户证书，建立在线的证书数据库系统。因此，其认证管理能力较低。同时，该方法仍然不适合网络世界中代理、进程和主体的认证。1999 年，中国著名的信息安全专家南相浩教授提出的基于标识的 CPK（Combined Public Key，组合公钥）算法开辟了新的天地。CPK 认证体制采用集中式管理模式，颁发的证书为 ID 证书，采用的是非第三方和非在线的认证形

式，适合网络世界中代理、进程和主体的身份认证，规模大、使用资源小。走向组合密钥产生体系是认证体系发展的要求，但是，组合后又会产生新的问题，即“存在共谋破解密钥体系的安全威胁”。

21.2 活性认证理念带来的变革

基于 CPK 密钥管理算法的优点及其存在的问题，笔者提出了活性认证的基本理论体系。活性认证基本包括四个方面：

- ◆ 使用活性密钥。
- ◆ 认证通过多代理技术体制实现，也称之为认证代理化。
- ◆ 提供主动认证服务。
- ◆ 活性认证还为 CPK 密钥管理算法提供了安全可信机制。

活性认证理念对于密码和认证行业来说是一个非常巨大的变革。通过活性认证在网络传输和处理中充分利用代理技术的活性特点、多代理技术组织性、多代理平台管理特性以及代理网格的群体计算能力，构建活性客体，建立活性密钥，实现活性认证技术体制，使活性密钥体系达到空前的安全与可信水平；同时，它从根本上改变了认证体系的模式、效率和质量。笔者在《软件行为学》一书中对现代新型的群体计算问题进行了详细的讨论，指出群体计算模型中会产生经典模型中所没有的新的能力，例如，在时空统一环境下的可生长、可移动的群体软件体系具有的无限计算资源。在网络时代，尤其在大范围网络环境中，处理海量数据和对象以及拥有海量的资源已经成为可能。笔者还指出这种群体计算具有不确定计算能力。传统计算的时间复杂性是指数性(2^n)

的问题，而在大范围网络环境中，将会变成低阶多项式复杂性问题。另外，群体计算在提高分析能力和群体生存能力方面具有的优势，也是传统的计算模型所不能比拟的。同时，尽管从理论上讲，不采用这种群体计算方法，行为与内容认证是可行的，但是在实际中仍然是不可行的。在《软件行为学》中，笔者还对软件行为控制、监管、认证和对抗等模式进行了详细的分析与研究，为活性认证的理论研究奠定了基础。另外，本书给出的“CPK 活性认证安全命题”是非常值得读者关注的。

国家组合公钥基础设施（National CPK Infrastructure, NCI）是利用 CPK 和活性认证技术建立的国家范围的信息化可信认证体系。笔者认为，NCI 可以成为代替 PKI 的计划。NCI 采用“CPK Inside”的核心技术服务模式，由以活性认证体系和可信认证为自主知识产权的面向应用服务的“NCI 企业联盟”和推进组合公钥理论与技术发展的“NCI 论坛”所组成。

可信活性认证在概念上做了较大的调整。

首先，认证的范畴发生了较大变化。不论在现实世界还是在网络世界中，认证不仅包括主体和客体的认证，还应当包括行为认证和内容认证两个范畴。

1) 主体认证

主体认证必须满足注册性(Re)、一体性(Ti)、解读性(Message Readable)要求，在应用模式 ρ 情景和鉴别状态 σ 条件下，构成下列函数：

$$F(S1)_{\rho\sigma} = (Re, Ti, MR)$$

2) 客体认证

客体认证公式在应用模式 ρ 情景和鉴别状态 σ 条件下为：

$$F(0) \rho\sigma = (\text{integrity, nonce, accountability of data})$$

其中，数据的完整性（integrity）证明数据完整无损；数据的新鲜性（nonce）证明本数据以前从没有发生过；数据的负责性（accountability）证明对数据负责的责任人。

3) 内容认证（统计意义上的相关性认证）

令内容 t 的前期内容记录为 $t_1 t_2 t_3 \cdots t_n$ ，在应用模式 ρ 情景和鉴别状态 σ 条件下，其鉴别公式的形式为：

$$F(C) \rho\sigma = \text{authenticity}(t_1 t_2 t_3 \cdots t_n t) \rho\sigma$$

4) 行为认证（统计意义上的相关性认证）

令行为 α 的前期行为踪迹为 $\alpha_1 \alpha_2 \alpha_3 \cdots \alpha_n$ ，在应用模式 ρ 情景和鉴别状态 σ 条件下，其鉴别公式的形式为：

$$F(\alpha) \rho\sigma = \text{authenticity}(\alpha_1 \alpha_2 \alpha_3 \cdots \alpha_n \alpha) \rho\sigma$$

其次，强调公众服务认证密钥体系是现代化公众服务体系的核心。建立以人为本的和谐社会，信息化的发展要特别强调建立大规模、高效、安全、可信、便捷的信息化公众服务体系。目前，我国各行业的企业独立建设的、规模较小的公众服务体系，在规模、效率、安全性、可信性和便捷性等方面都存在着普遍的问题。第一，整个公众服务体系必须实现统一的总体要求，统一用户使用方式与界面，实现公众服务系统的互联、互通和互操作。第二，要求确保公众服务体系的安全与可信。要对公众服务的信息内容诈骗、行为侵害、用户隐私泄露、出卖客户隐私信息、损害客户权益、企业雇员与社会犯罪集团内外勾结的事件实行法律、行政和技术多种手段的防范与打击，尤其要强化信息化的监管与控制手段。第三，要大力提高公众服务企业雇员的安全意识、素质以及行业的公信力。

我国密钥体制是从通信密码体系延续过来的，实行的是国家安全意义上的密码管理体制。笔者认为，应当改革我国密钥管理体制，尤其要建立为百姓服务的公众密钥体系。

那么，如何实现活性认证呢？我们首先定义活性客体的概念：

令活性客体为 $X[O]$ ，它是 $\text{ActiveObject} = \{(X, O) \mid X \text{ 是一个代理}, O \text{ 是一个客体}\}$ 中的对象。这个 $X[O]$ 中的 X 与 O 具有如下特性：

- ◆ 如果客体 O 被打开或被操作，那么代理 X 同时被创立与激活；如果客体被关闭或停止对其操作，那么代理 X 自行终止执行。
- ◆ 代理 X 被创立与激活，客体 O 可以被打开或不被打开。
- ◆ 如果客体 O 被其他主体操作，那么代理 X 可以感知并做出相应的反应（例如，消除客体 O 、定位被劫持设备与系统）。
- ◆ 代理 X 执行被其他主体终止，客体 O 将被消除。

活性客体是一个统一的实体，如果用通常的类型表达式来表示（即 $\text{ActiveObject} = \text{Agent} \times \text{Object}$ ），那么虽然反映出了组合概念，但反映不出一个实体的不可分割的概念。如果一个代理与某个信息客体通过实现机制在信息系统中传输、处理、存储都被视为统一的实体，那么这个实体是活性的，是能感知指定功能和做出相应操作并与客体伴随的统一实体。

在此基础上，进一步提出活性密钥的概念：

令活性密钥为 $X[k]$ ，它是 $\text{ActiveObject} = \{(X, k) \mid X \text{ 是一个代理}, k \text{ 是一个密钥或产生密钥的对象，包括密钥产生算法和产生矩阵等}\}$ 中的对象。我们把活性密钥的代理称做活性密钥代理或活性认证代理。这个 $X[k]$ 中的 X 与 k 除具有活性客体的基本特性之外，还具有如下三个应用要求特性：

- ◆ 如果代理 X 的主人发出认证的要求，那么代理 X 将根据主人的指令提供认证服务。
- ◆ 代理 X 具有构成有组织群体的机制和能力，即构成活性密钥代理的集合，并组成活性密钥代理的网络，从而构成一个认证的多代理系统。认证多代理系统中的代理之间能够感知相互存在和状态，并记录群体运营的相关信息。认证多代理系统中的所有认证代理接受统一配置和管理的服务要求。
- ◆ 认证代理具有可生长和可移动要求。

活性密钥是一种新型的实体结构。在传统的系统概念中，把实体划分为主体（进程等）和客体（密钥等）两种类型。而活性客体既不是主体，也不是客体，但同时保留了主体和客体两面的属性。现代计算机的操作系统还没有这种新型的活性客体的分类实体，我们只有期待未来的计算机和操作系统中设计这种新的实体结构（目前，可以用主体中私有客体来替代）。

在上述概念基础上建立活性认证代理和执行认证的有组织的多代理系统和多代理管理平台。大规模认证需要提供组合密钥算法，利用一个组合密钥矩阵产生巨大规模的公钥和私钥，是认证发展的必然要求。组合自身也会带来安全问题，例如采用组合公钥体系，私钥体系分发的安全就是一个关键问题：攻击者以合法和非法的形式掌握大量用户私钥，通过共谋方法破解密钥体系。为了解决安全问题，应实行如下的安全策略：

- ◆ 活性客体安全策略
- ◆ 生存期管理安全策略
- ◆ 多代理群体组织的安全策略

- ◆ 异常处理的安全策略
- ◆ 对认证工作实行监管的安全策略

通过上述 5 个方面的策略讨论，可以说活性密钥在活性世界中应当被认为是安全的，并有如下的安全命题：

CPK 活性密钥体系充分利用代理技术特点、多代理技术组织性、多代理平台管理特性以及代理网络的群体计算能力，构建活性客体，建立活性密钥，实现活性认证技术体制。在活性密钥能够存活的世界中，使活性认证体系的密钥（除密钥分发中心外），可以被使用而永不驻留在传输、处理的任何设备之中，从而在该世界中不可能获得认证密钥体系。攻击者企图利用其他手段获得活性密钥体系并重新进入活性认证代理系统发挥作用是可发现、可阻击的。

实际上，CPK 活性认证体系的提出基于如下层次结构的安全思想：

- ◆ 首先，在正常环境中，密钥（主要指私钥）可用而不能拥有和获得，这就是密钥活性化的意义。
- ◆ 其次，如果密钥通过非常手段被拥有或获得，那么该密钥不能回到原系统中来。也就是说，每一个活性密钥都有个体和群体两个完整性，而且这种群体完整性是与系统中的群体状态相关的，通过群体完整性来进一步确认其安全。另外，需要说明的是，这两个完整性是系统强制的，而不是用户与管理员自主的。
- ◆ 第三，如果这个退出系统的密钥非要回到系统中来，那么它将被认为是“陌生”的，并被发现和阻击。也就是说，通过各种管理来维护活性密钥的两个完整性。

读者可以设想上述的发展思路：从密钥是一个非活性的客体到把密钥活性化，从而实现密钥可用而不能获得；又从活性密钥的个体发展到活性密钥的群体，并通过动态实现群体完整性来保证被截获的活性密钥个体不能返回系统群体。攻击者必须同时得到一个系统中的全部活性密钥，并在系统群体活性密钥完整性发生强制变化前（很短时间内）实现截获和返回系统，这实际上是不可能的。

21.3 行为认证形式化描述

行为认证问题在形式化描述方面可以有如下的讨论：

行为认证是整个认证领域中取得可信性与有效性的重要组成部分，是身份认证发展阶段的新问题。《软件行为学》一书中做了如下的行为认证本体语义描述：

对于行为表达式 $\alpha_1 \Rightarrow \alpha_2$ （其中， α_1 表示被认证行为， α_2 表示认证行为），其行为语义为：

$$\alpha_2 = \lambda(\alpha). (S_2 \langle S_1 \rangle : \text{AUTHENTICATE}(\alpha))$$

其中，AUTHENTICATE: $A \rightarrow F \rightarrow \text{ENV} \rightarrow \text{ST} \rightarrow \text{ST}$ 监管函子的类型，A表示行为信息，F表示认证方法，ENV表示认证环境，ST表示认证状态。下面，我们举一个例子来说明这种形式化方法的意义。

令 $S = \{S_0, S_1, S_2, \dots, S_n\}$ 表示网络中的一个业务主体集合，其组织关系可以表示为 $S_0 :: S_1 S_2 \dots S_n$ ；其中， S_0 是管理者，而 S_1, S_2, \dots, S_n 是被管理的主体。 $A = \{A_0, A_1, A_2, \dots, A_n\}$ 表示一个认证代理的集合，其组织关系可以表示为 $A_0 :: A_1 A_2 \dots A_n$ ，

是 $S_0, S_1, S_2, \dots, S_n$ 的认证代理的组织形式, 与 $S_0 :: S_1 S_2 \dots S_n$ 结构一一对应。但是, 认证代理与业务主体没有构成固定的伴侣关系。认证代理是一个可生长和可移动的有组织群体多代理系统。

令 $K = \{K_1, K_2, \dots, K_n\}$ 表示认证的密钥对集合, 即 $K_i = (ski, PK)$, ski 表示私钥, $i \in \{1, 2, \dots, n\}$, PK 表示 CPK 体系的公钥矩阵; 再令 $ActiveKeyAgentSet = \{ActiveKeyAgent_0, ActiveKeyAgent_1, ActiveKeyAgent_2, \dots, ActiveKeyAgent_n\}$ 表示另一个活性密钥代理的集合, 其组织关系可以表示为 $ActiveLabel_0 :: ActiveLabel_1 ActiveLabel_2 \dots ActiveLabel_m$ 。其中, $ActiveKeyAgent_1, ActiveKeyAgent_2, \dots, ActiveKeyAgent_n$ 与客体 K_1, K_2, \dots, K_n 构成一一对应的伴侣形式, 即有如下的活性密钥体系: $ActiveKeyAgent_1[K_1], ActiveKeyAgent_2[K_2], \dots, ActiveKeyAgent_n[K_n]$ 。所以, $ActiveKeyAgent_0 :: ActiveKeyAgent_1[K_1], ActiveKeyAgent_2[K_2], \dots, ActiveKeyAgent_n[K_n]$ 。

在设计活性认证系统时, 可以考虑将认证代理体系和活性密钥代理合并成一个多代理系统。当然, 也可以设计成一组代理, 它们之间有一一对应的伴侣关系, 即: $A_1 < ActiveKeyAgent_1[K_1] >, A_2 < ActiveKeyAgent_2[K_2] >, \dots, A_n < ActiveKeyAgent_n[K_n] >$ 。

令 γ_i 是主体 S_i 的组织行为, $i \in \{1, 2, \dots, n\}$, 并有 $\gamma_0 = S_0: Management(\gamma_1 \parallel \gamma_2 \parallel \dots \parallel \gamma_n)$ 。

令 χ_i 是认证代理 A_i 的行为, $i \in \{1, 2, \dots, n\}$, 并有 $\chi_0 = A_0: Management(\chi_1 \parallel \chi_2 \parallel \dots \parallel \chi_n)$ 。

令 α_i 是主体 S_i 的业务行为, $i \in \{1, 2, \dots, n\}$, 并有 $\alpha_0 = S_0: Management(\alpha_1 \parallel \alpha_2 \parallel \dots \parallel \alpha_n)$ 。

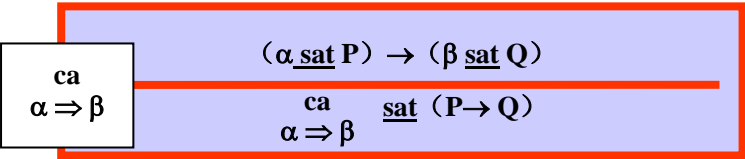
令 β_i 是认证代理 A_i 的认证行为, $\beta_i = \lambda(\alpha).(A_i < S_i >: AUTHENTICATE(\alpha))$, $i \in \{1, 2, \dots, n\}$, 并有 $\beta_0 = A_0: Management(\beta_1 \parallel \beta_2 \parallel \dots \parallel \beta_n)$ 。

$\alpha_0 \Rightarrow \beta_0$ 表示认证代理对业务主体的行为 α_0 实行认证行为 β_0 ，并且有如下关系：

$$\begin{aligned} \alpha_0 \Rightarrow \beta_0 &= A_0 \langle S_0 \rangle : \text{AUTHENTICATE}(\alpha_0) \\ &= (\alpha_1 \Rightarrow \beta_1 \parallel \alpha_2 \Rightarrow \beta_1 \parallel \cdots \parallel \alpha_n \Rightarrow \beta_1) \\ &= ((A_1 \langle \text{ActiveKeyAgent}_1[K_1] \rangle) \langle S_1 \rangle : \\ &\quad \text{AUTHENTICATE}(\alpha_1) \\ &\quad \parallel (A_2 \langle \text{ActiveKeyAgent}_2[K_2] \rangle) \langle S_2 \rangle : \\ &\quad \text{AUTHENTICATE}(\alpha_2) \\ &\quad \parallel \cdots \cdots \\ &\quad \parallel (A_n \langle \text{ActiveKeyAgent}_n[K_n] \rangle) \langle S_n \rangle : \\ &\quad \text{AUTHENTICATE}(\alpha_n)) \end{aligned}$$

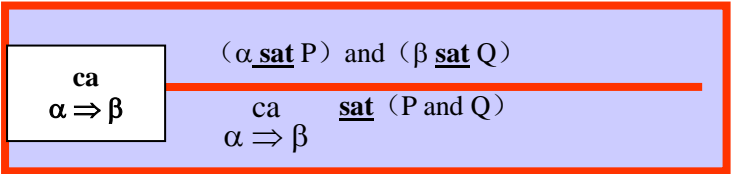
行为认证有以下两种模式：

◆ **非控制型认证** 行为认证不干预被认证行为的过程，没有控制作用，仅仅通过认证颁布一种评价证书。行为认证的逻辑语义描述如下：如果利用逻辑思考的原理，那么只有在被认证行为 α 可信（满足条件 P ）时， β 可信和有效（满足条件 Q ）， $\alpha \Rightarrow \beta$ 才可信和有效（满足条件 $P \rightarrow Q$ ）。如果被认证行为是不可信的（不满足条件 P ），认证方法（评估方法）是真实的，那么认证依然是虚假的（欺诈、假票据、假账等），所以 $\alpha_1 \Rightarrow \beta$ 应当满足的条件是 $P \rightarrow Q$ ，如下所示：



◆ **控制型认证** 行为认证干预被认证行为的过程，有控制

作用，同时，颁布认证书。行为认证的逻辑语义描述如下：如果利用逻辑思考的原理，那么只有在被认证行为 α 可信（满足条件 P），同时 β 可信和有效（满足条件 Q）的情况下， $\alpha \Rightarrow \beta$ 才是可信和有效的（满足条件 P and Q）。如果被认证行为是不可信的（不满足条件 P），或认证方法（评估方法）是不真实的（不满足条件 Q），那么认证是虚假的（不满足条件 P and Q），如下所示：



第 22 章

网络行为对抗理论

22.1 建立网络对抗的威慑力量

网络对抗理论是网络行为学中的一种行为应用模式理论。建立网络对抗的威慑力量是信息化安全的重要领域。安全威慑体系主要针对有组织犯罪、有较大和很大资源的攻击者对我国基础设施信息系统发出的攻击威胁，通常意义上的防护或保障体系已经不能防止这种威胁的发生，提供对攻击者实施打击的力量和能力是非常必要的，让任何攻击者在实施攻击时都要三思而行，不会轻易攻击。这从另一个方面说明了企业或行业安全不仅仅是这些部门的事情，还是国家，尤其是国家政法部门（例如公安、安全、保密、司法、检查、法院等部门）和军事部门共同的事情，在必要时，它们都能用得上。

我们归纳了如下三种威慑力量：

- ◆ 第一种威慑力量是打击公共信息网络的犯罪行为的力量。因为网络犯罪和破坏是人在计算机中的代理软件完成的，所以防护、打击和执法也必须在网络中通过其代理完成。我们无法设想：犯罪和破坏在网络内，而防护、打击和执法在网络外。
- ◆ 第二种威慑力量是打击信息恐怖主义的力量，其主要任务是打击国家要害行业信息专网上的有组织犯罪行为和恐怖主义行为。
- ◆ 第三种威慑力量是在战争中实施网络对抗的力量。所谓网络对抗的战争形态，是指双方建立一支多种功能代理的作战协同体系，建立网络中的虚拟组织和军队。双方有攻击，也有防护。在很大程度上，消灭对方的网络代理系统是网

络对抗的重要内容，有时甚至是主要内容。

建立打击网络犯罪的数字化警察部队，建立打击网络敌对势力与恐怖势力的数字化安全部队，建立能够实施网络对抗的数字化师或数字化军，是三个独立发展的国家主权意义上的威慑体系建设。它们之间相互依存、不可替代。

发达国家对国防信息化中的网络对抗十分重视。美军在 2000 年进行了一次针对金融的网络对抗战。从国外的国防信息化的网络对抗中，我们可以归纳出如下三方面的建设内容：

1. 国防网（或称军网）的自身安全建设（应当清楚，虽然国防网是专网，但是从战争与反恐的角度来看，它是非封闭的，而是开放的），要从保障、应急两个方面去努力建设。

2. 军网对抗的军事威慑力量建设，研究网络对抗的技术、装备与战术，而不仅仅是采用一些黑客的攻防技术。网络对抗是双方的多种功能代理的作战协同体系的对抗。所谓网络对抗，是通过真实作战人员的直接对抗操作或主要通过各自代理系统向对方系统和对方网络代理系统发起攻击。对抗可以实施多种战术，战术优劣仍然是决定胜负的重要因素。双方有攻击，也有防护。在很大程度上，消灭对方的网络代理系统是网络对抗的重要内容，有时甚至是主要内容。双方产生网络作战代理的能力，不断补充被消灭的网络代理系统是决定胜负的关键因素。网络对抗战争威胁的主要特点是：

- ◆ 具有一定的打击能力、破坏能力和打击范围。
- ◆ 精确可控（包括敌我识别、武器可控、能力可控、范围可控、过程可控等。例如，计算机病毒不可控，故不能称为武器）。

- ◆ 能实施一定战术，协同工作，可移动。
- ◆ 武器平台化，成为装备。

3. 重视国防力量对网络“国土防护”的安全保卫力量的建设。加强军方对电信、金融、电力等网络基础设施的防护和对敌方基础设施网络安全威慑力量的建设，参与研究国家网络国土防护的战术、技术、装备和人员训练等一系列新问题。

22.2 网络行为对抗理论

下面，简单介绍一下《软件行为学》一书中关于行为对抗模式的讨论。

网络世界中的虚拟主体相互之间针对对手的生存性、行为控制、行为特性的行为活动，称为网络行为对抗。行为对抗必须是相互的，只要有一方放弃对抗，另一方便失去了对抗的意义。严格地讲，通常见到的黑客攻击和安全防护之间的关系，构不成对抗模式。因为通常的防护虽然在概念上是针对攻击的，但是攻击来了也只有检测和防护行为，并没有直接的对抗。行为之所以可以或能够对抗，是由于存在着两种行为，它们的功能是完全相反的。在网络虚拟世界中，同样可以找到与人类社会世界和物理世界中存在的行为对抗概念相似或相同的对抗概念。行为对抗可以分为个体（包括代理）行为对抗和群体（包括多代理）行为对抗。

网络对抗是对抗体系、对抗理论的对抗。对抗理论模型的好坏会影响网络对抗的成败。对抗理论模型并不是固定的某种数学模型，而是建立在软件行为学基础上的一种对抗双方在某个环境内相生相克的模型。软件行为理论就是软件行为对抗理论，这个

理论直接指导软件行为的对抗活动。

网络对抗是對抗体系的运行管理体系结构的对抗。对抗体系的运行管理体系结构的好坏会影响网络对抗的成败。

网络对抗是人与其代理结合的对抗，必须建立一个好的对抗代理体系、一个方便和有效的运行管理模式以及一个高效的运营体系结构。

网络对抗是對抗体系的系统体系结构的对抗。有了好的理论模型和运营体系结构，还需要有好的系统体系结构。对抗的系统体系结构的好坏会影响网络对抗的成败。

网络对抗是對抗体系的技术体系结构的对抗。对抗的技术体系结构的好坏同样会影响网络对抗的成败。

对抗体系的部件和组成采用同样的技术标准是非常重要的。

另外，网络对抗是對抗体系的对抗战术的对抗，对抗战术的好坏也会影响网络对抗的成败。

行为对抗模式分为三大类：

- ◆ 生存性对抗
- ◆ 控制性对抗
- ◆ 行为特性对抗

生存性对抗是指以对手的生存性为对抗目标的行为对抗，对抗的目的在于发现对手并消除对手。这种对抗可以是针对对手执行含义的消除，但是对手的执行文件依然存在；这种对抗还可以既针对生存性，又针对执行文件存在。群体中个体的生存性与群体的生存性是两个概念。群体的生存性实际上建立在群体中个体的克隆特性的基础之上。

控制性对抗以控制对手为目的，控制对手的行为本身。这种控制是什么呢？主要是控制群体中个体的协同能力和协同状态，

控制个体的活动能力和应用操作能力，例如，个体总在从事无效的行为。

行为特性对抗以控制对手行为的某些特性为目的。例如，针对行为保密性、完整性等特性进行对抗。行为保密性对抗的目的不是让对手不生存、不活动，而是期望在对手的活动中，了解对手行为的信息和特点；行为完整性对抗的目的在于参与或代替对手的行为活动，破坏对手行为的完整性。

行为对抗组织结构与行为对抗的目标和环境密切相关。行为对抗组织是人-代理交互的系统结构，其中包括人如何与代理交互，需要多少人参加行为对抗组织等。如何建立代理群体组织是行为对抗组织第一位的问题。建立对抗组织的第一个问题是：建立对抗组织的结构，这种结构的建立是根据对抗的任务要求进行的。建立一个有效的对抗组织是实现行为对抗的必要条件。建立对抗组织的第二个问题是：确定组织中个体之间的协同关系和协同性质，要求实现可靠的协同联系。建立对抗组织的第三个问题是：研究组织中每一个个体的结构与功能，要求对组织中个体的结构和功能进行业务与强度的分类，论证组织中需要什么样的个体业务和什么强度水平的功能。建立对抗组织的第四个问题是：研究对抗组织内部的管理、识别、认证和安全等。

从理论上讲，行为对抗的模式是任意的。从行为的目的来看，有正面的行为，就有反面的行为（逆行为）；从行为的结果来看，同样也可以找到相互对立的行为组合。但是，我们要研究的对抗行为是指能够持续的代理有组织群体的对抗，那种没有后续能力的一次性行为对抗不是我们要研究的模式。不考虑对抗的环境背景，仅仅从对抗的目的性考虑，在行为对抗的三个模式中，本书主要研究如下的问题：行为对抗能力、行为对抗资源、行为对抗策略、行为对抗战术、行为对抗技术、行为对抗支持技术、行为

对抗平台与系统。

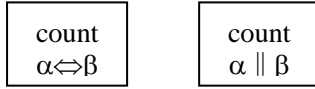
网络对抗技术包括：代理生存与消除对手技术、模式发现与模式隐藏技术、行为发现与行为隐藏技术、行为控制与反控制技术、行为特性对抗技术、攻击入侵技术（例如“黑客”模式攻击方法、“战争”模式寄寓“和平”工作模式之中的攻击方法、快速有线插播攻击方法、卫星通信攻击方法、无线通信攻击方法等）、定位与反定位技术、追踪与反追踪技术、行为对抗组织输送和配置技术、行为对抗能力评估（红/蓝测试）等。

可以将网络行为对抗学作为专门的学问来研究，例如，可以称为“网络行为对抗学”。网络行为对抗的任务是打击网络犯罪、打击网络恐怖主义和在网络上实现信息战对抗。在人类社会中，打击犯罪和战争已经有几千年的历史了；在网络中，打击犯罪和信息战才刚刚开始。网络虚拟世界中的战争是什么样子？网络虚拟世界中的军队是什么？虚拟军队的组织和行为特点是什么？还没有可供借鉴的经验和理论。本书主要探索如果发生信息战网络对抗，计算机系统和通信系统能够做些什么。

《网络行为对抗学》是笔者正在撰写的书籍，包括网络行为对抗基本概念、网络行为对抗模式、网络行为对抗逻辑学、网络行为易理逻辑学（对抗态势演化）、网络行为对抗博弈理论、网络行为对抗平台、网络代理执法学、网络行为保密学、数字警察学、数字隐蔽对抗和信息战网络对抗学等内容，希望不久后能够与读者见面。

22.3 行为对抗形式化描述

我们可以用如下的符号来描述行为对抗：

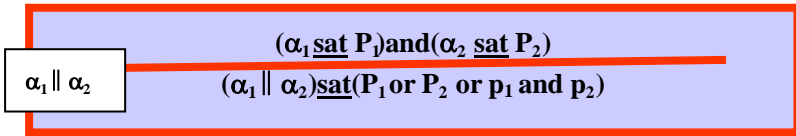


但是为了方便，只要不会发生混淆，我们依然用 $\alpha_1 \parallel \alpha_2$ 表示行为对抗，其语义为：

$$S_1: F_1(\alpha_2) \parallel S_2: F_2(\alpha_1) \text{ where } \alpha_1 = \lambda(\alpha).(S_1: F_1(\alpha)), \alpha_2 = \lambda(\alpha).(S_2: F_2(\alpha))$$

软件行为对抗的逻辑语义描述如下：

行为对抗模式是群体组织的业务功能的行为模式。采用 $\alpha_1 \parallel \alpha_2$ 表示，行为 α_1 和 α_2 之间的关系是任意的，除了可以规定它们之间是并发的，不能预先规定任何其他特性。



如果两个行为的共同结果使得 $P_1(X)$ 满足，那么上述行为对抗中 α_1 达到目标，记为 $(\alpha_1 \parallel \alpha_2) \text{ sat } P_1(X)$ ；如果两个行为的共同结果使得 $P_2(X)$ 满足，那么上述行为对抗中 α_2 达到目标，记为 $(\alpha_1 \parallel \alpha_2) \text{ sat } P_2(X)$ ；如果两个行为的共同结果使得 $P_1(X)$ 中的子项 $p_1(X)$ 满足，同时使得 $P_2(X)$ 中的子项 $p_2(X)$ 满足，那么称上述行为对抗部分达到目标，记为 $(\alpha_1 \parallel \alpha_2) \text{ sat } p_1(X) \text{ and } p_2(X)$ ，其中， $p_1(X)$ 是能够满足的最大 $P_1(X)$ 子项， $p_2(X)$ 是能够满足的最大 $P_2(X)$ 子项。

22.4 网络虚拟军队主体行为结构描述

为了便于读者理解行为对抗理论与方法，我们通过一个例子来讨论对抗组织结构和组织对抗行为模式。这里谈到的行为对抗组织是指代理组织，可以划分为数字化军、数字化师、数字化团、战斗组和战斗员。这些组织对抗行为模式的差别在什么地方呢？下面，我们用方案举例的方法来描述网络行为对抗的行为模式设计：

- ◆ **数字化军对抗行为模式** 数字化军是依据一个国家的网络对抗设计定位的，主要负责基础设施网络的对抗事项，负责各数字化师协同，接受数字化师的请示，对数字化师的对抗任务进行指示，同时有创立、输送和配置数字化师的功能。数字化军没有自我克隆的功能。数字化军有一个可视化的人工处理系统，在它的后面还有一个军指挥官代理系统，可以在人工干预下工作，也可以在无人值守的情况下工作。
- ◆ **数字化师对抗行为模式** 数字化师被设计定位为负责一个行业之内的跨企业的各数字化团的对抗行为的协同。数字化师的工作范围被严格限制在行业网络范围之内，负责创立、输送和配置数字化团到指定位置。数字化师本身没有处理任务，接受各数字化团的请示并向数字化团下达业务指令和协同指令。数字化师有向上级数字化军请示的责任。数字化师没有自我克隆的功能。数字化师有一个可视化的人工处理系统，在它的后面还有一个师指挥官代理系

统，可以在人工干预下工作，也可以在无人值守的情况下工作。

- ◆ **数字化团对抗行为模式** 数字化团被设计定位为执行一个企业范围内的对抗任务，这个企业可能在广域网或区域网范围内。数字化团的任务是在工作活动范围内创立、输送和配置战斗组到企业范围网络的各结点上。数字化团自身没有处理任务，负责协同团内各战斗组的对抗活动，接受战斗组的请示并下达行动的指令，例如，允许某些战斗组进行克隆，来扩大战斗组的数量。数字化团必须向上级数字化师或直接向数字化军进行业务请示，接受上级数字化师或数字化军的指令。数字化团能够以团为单位进行团之间的行为协同。数字化团没有自我克隆的功能。数字化团有一个可视化的人工处理系统，在它的后面还有一个团指挥官代理系统，可以在人工干预下工作，也可以在无人值守的情况下工作。
- ◆ **战斗组对抗行为模式** 战斗组由数字化团创立、输送和配置到指定网络结点的计算机上。战斗组由组长领导，有克隆、防护、发现和攻击四种重要功能。战斗组的每一个代理都与组长代理进行协同，也可以在组员之间进行协同。战斗组可以在上级数字化团的指示或许可下以全组为单位进行克隆，可以以组为单位进行组间协同和业务联系。
- ◆ **战斗员（战士、士兵）对抗行为模式** 战斗组中处理代理的数量可以随着战斗任务的变化而变化，一个战斗组的工作范围通常限制在一台计算机上或一个局域网内，不能离开规定的活动范围。可以设计克隆代理(c)、防护代理(p)、

检测代理(d)和攻击代理(a)。克隆代理专门承担克隆复制任务；防护代理专门承担防护任务；检测代理专门承担发现任务，主要目的在于发现目标对象；攻击代理是实施攻击的代理。

行为对抗必须包括如下信息：对抗能力、态势、战术预案和策略等。在对抗中，应当提供策略服务（目标确立、攻击与防护模式）、战术编辑服务（编辑战术方案）、战术预案服务（准备、演练、改进战术预案）、能力配置服务（为战术预案配置数字化军事建制）、资源后勤服务（研发新的攻击与防护体系）、指挥调度服务（发出攻击、防护、输送和停止命令等）、输送服务（为输送攻击与防护代理组织到指定位置提供服务）、攻击服务（实施预案规定的攻击）、防护服务（实施预案规定的防护）。对抗组织应当具有预警能力、调度/隔离能力、恢复能力以及提供新型的攻击、防护、发现、克隆品种和新型的种属组代理仓库能力等。对抗态势是由指挥官代理状态和下属指挥官代理态势表所组成的。指挥官代理的状态采用代理组织的协同与业务状态来定义，其业务状态可以使用“混沌”、“存在”、“预备”、“战斗”、“中断”、“发生异常”、“胜利终止”、“失败终止”等状态标识来表示。应当明确数字化军、数字化师、数字化团和战斗组的对抗战术原则，通常包括若干行为对抗作战方案、演习方案、无人值守方案、输送方案、配置方案、人员控制方案等。要有攻击目标表、防护目标表、攻击模式、防护模式、攻击目标地址表等属性。

实现网络对抗，要建立网络行为对抗平台体系。网络行为对抗平台体系是一个可视化系统与多代理系统相结合的体系结构。一个信息战的操作功能系统采用的主要技术是多代理，但是决策、

调度、指挥、管理、配置、后勤支持等要依托在可视化的信息系统之上，而不能一味地追求全代理化。下面，我们以数字化师为例来说明网络行为对抗的原理。数字化师由可视化系统和数字化师指挥官代理所组成。由于数字化师系统的主要任务是决策、调度、指挥、管理、配置、后勤支持等，所以应当以人控制的可视化系统作为主要的运作方式，以师指挥官代理作为执行或无人值守的系统部件。IW 师可视化系统可以有相当规模的计算机与通信系统作为支持环境，与所有的 IW 团可视化系统连接，接受团可视化系统的信息或向团可视化系统发出命令。师指挥官代理同时与团指挥代理连接，接受团指挥官代理的信息或向团指挥官代理发出命令。也就是说，师与团之间有两个连接通路：可视化系统之间的连接和相应指挥官代理之间的连接。师可视化系统可以在固定环境下运行，也可以在车载移动环境下运行。数字化师是国家范围内某行业领域（例如金融、电信、电力、政府、国防等）概念上的网络威慑力量体系结构（如图 22-1 所示），数字化团是企业级的信息化威慑力量。团以下的 IW 力量没有可视化系统与之对应运行，是全代理系统的体系结构。从上面的讨论中可以看出，在军、师、团三个层次上存在着可视化系统。多代理系统承担着操作层面的重要任务，是可视化系统不可替代的。

假定在一个国家建立一个网络对抗数字化军，这个数字化军有 10 个数字化师，每一个数字化师负责保卫一个国家基础设施网络体系（金融、电力、电信、政府等），每一个数字化师有 100 个数字化团，每一个数字化团有 60 个战斗组，每一个战斗组有 4 个战斗代理，其结构如图 22-2 所示。当然，对每一个基础设施网络国土的防卫是不相同的，但是这里为了说明原理，假定每一个数字化师、数字化团和战斗组都是相同类型的。

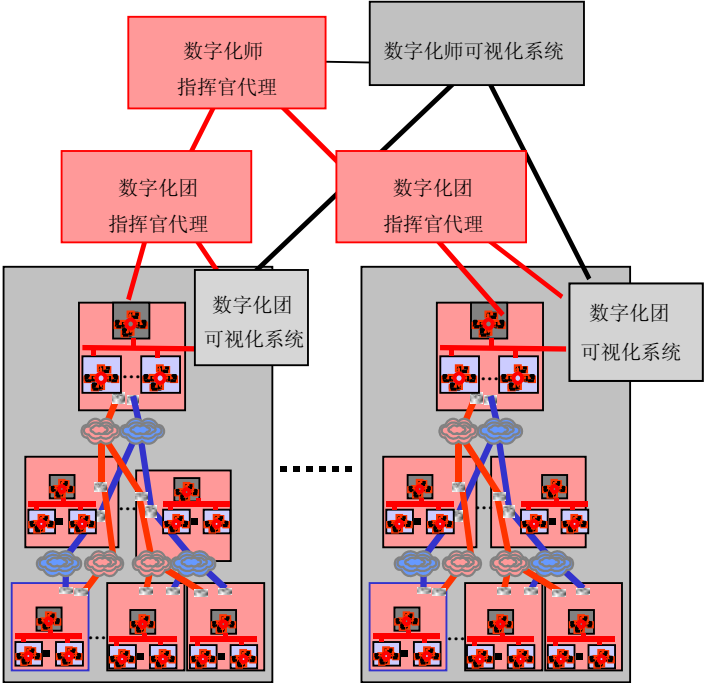


图 22-1 行业领域网络行为对抗平台

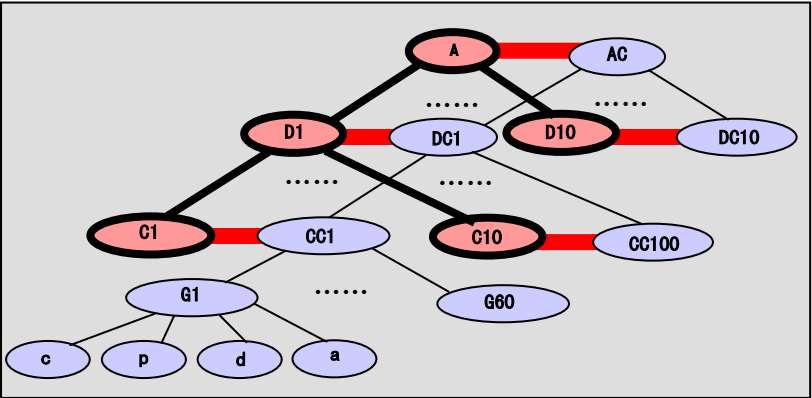


图 22-2 行为对抗主体结构图

在图 22-2 中, A 表示一个数字化军的可视化系统, 而 AC 表示数字化军的指挥官代理; D 表示数字化师的可视化系统, 而 DC 表示数字化师的指挥官代理; C 表示数字化团的可视化系统, 而 CC 表示数字化团的指挥官代理; G 表示战斗组, 而下面的 c 表示克隆代理, p 表示防护代理, d 表示检测代理, a 表示攻击代理。下面, 分别对红蓝双方定义对抗组织的代理结构。红方组织有:

RAC (RDC)¹⁰----- 红方数字化军的指挥官代理指挥 10 个数字化师的指挥官代理
 RDC (RCC)¹⁰⁰----- 红方数字化师的指挥官代理指挥 100 个数字化团的指挥官代理
 RCC (RG)⁶⁰----- 红方数字化团的指挥官代理指挥 60 个战斗组组长代理
 RG cpda----- 红方战斗组组长代理组织与指导 cpda 4 个对抗功能代理

红方组织在行为管理模式上有如下的公式:

$\alpha_A \Leftrightarrow \alpha_{AC} = \text{RAC}: \text{RA_Command}(\alpha_{DC1} \parallel \alpha_{DC2} \parallel \cdots \parallel \alpha_{DC10})$
 $\alpha_D \Leftrightarrow \alpha_{DC} = \text{RDC}_i: \text{RD_Command}_i(\alpha_{CC1} \parallel \alpha_{CC2} \parallel \cdots \parallel \alpha_{CC100}), 1 \leq i \leq 10$
 $\alpha_C \Leftrightarrow \alpha_{CC} = \text{RCC}_j: \text{RC_Command}_j(\alpha_1 \parallel \alpha_2 \parallel \cdots \parallel \alpha_{60}), 1 \leq j \leq 100$
 $\alpha_{G0} = \text{RG}_k: \text{RG_Direct}_k(\alpha_c \parallel \alpha_p \parallel \alpha_d \parallel \alpha_a), 1 \leq k \leq 60$
 $\alpha_G = \text{RG}_k: \text{RG_Clone}_k(\text{RG}), 1 \leq k \leq 60$
 $\alpha_c = \lambda(\text{agent}).(\text{Rc}: \text{R_clone}(\text{agent}))$
 $\alpha_p = \lambda(\beta).(\text{Rp}: \text{R_protection}(\beta))$
 $\alpha_d = \lambda(\beta).(\text{Rd}: \text{R_detection}(\beta))$
 $\alpha_a = \lambda(\beta).(\text{Ra}: \text{R_attack}(\beta))$

蓝方组织有:

BAC (BDC)^N----- 蓝方数字化军的指挥官代理指挥 N 个数字化师的指挥官代理
 BDC (BCC)^M----- 蓝方数字化师的指挥官代理指挥 M 个数字

化团的指挥官代理
BCC (BG)^K----- 蓝方数字化团的指挥官代理指挥 K 个战斗
组组长代理
BG cpda----- 蓝方战斗组组长代理组织与指导 cpda 4 个
对抗功能代理

蓝方组织在行为管理模式上有如下的公式：

$$\begin{aligned} \beta_A &\Leftrightarrow \beta_{AC} = BAC: BA_Command(\beta_{DC1} \parallel \beta_{DC2} \parallel \cdots \parallel \beta_{DC10}) \\ \beta_D &\Leftrightarrow \beta_{DCi} = BDC_i: BD_Command_i(\beta_{CC1} \parallel \beta_{CC2} \parallel \cdots \parallel \beta_{CC100}), 1 \leq i \leq N \\ \beta_C &\Leftrightarrow \beta_{CCj} = BCC_j: BC_Command_j(\beta_1 \parallel \beta_2 \parallel \cdots \parallel \beta_{60}), 1 \leq j \leq M \\ \beta_{GOk} &= RG_k: BG_Direct_k(\alpha_c \parallel \alpha_p \parallel \alpha_d \parallel \alpha_a), 1 \leq k \leq K \\ \beta_{GRk} &= RG_k: BG_Clone_k(BG), 1 \leq k \leq K \\ \beta_c &= \lambda(agent).(Bc: B_clone(agent)) \\ \beta_p &= \lambda(\alpha).(Bp: B_protection(\alpha)) \\ \beta_d &= \lambda(\alpha).(Bd: B_detection(\alpha)) \\ \beta_a &= \lambda(\alpha).(Ba: B_attack(\alpha)) \end{aligned}$$

有了上述的红篮双方的对抗组织之后，可以描述它们的对抗行为方案。假定一个指挥官可以提供如下的行为对抗服务（其实它们都是一些函子）：

$\lambda(\delta).Count(\delta)$ ----- 对抗服务(对抗的总体服务)
 $\lambda(\chi).Police(\chi)$ ----- 策略服务(目标确立、攻击与防护模式)
 $\lambda(\gamma).Tactics:Edit\lambda(\gamma)$ ----- 战术编辑服务(编辑战术方案)
 $\lambda(\alpha,\beta).Tactics(\alpha,\beta)$ ----- 战术预案服务(准备、演练、改进战术预案)
 $\lambda(\gamma,s).CapabilityConfig(\gamma,s)$ --- 能力配置服务(为战术预案配置对抗组织)
 $\lambda(\chi).Develop(\chi):$ ----- 资源后勤服务(研发新的攻击与防护体系)

$\lambda(\alpha).Command(\alpha)$ -----指挥调度服务(发出各种命令)
 $\lambda(\alpha,\beta).AnalysisSituation(\alpha,\beta)$ ---态势分析服务(对抗态势的分析服务)
 $\lambda(\chi,s).Transportation(\chi,s)$ ---输送服务(为输代理组织到指定位置提供服务)
 $\lambda(\delta).Attack(\delta)$ -----攻击服务(实施预案规定的攻击)
 $\lambda(\delta).Protection(\delta)$ -----防护服务(实施预案规定的防护)
 $\lambda(\delta).Information(\delta)$ -----情报服务(搜集对抗者的情报的服务)

例如,红蓝对抗可以描述为: $RAC:count(\beta_{AC}) \parallel BAC count(\alpha_{AC})$, 在这样的环境中,依据对抗预案(见图 22-3)进行实施。

$$RAC: \begin{pmatrix} \alpha_{11} & \alpha_{12} & \cdots & \alpha_{1n} \\ \alpha_{21} & \alpha_{22} & \cdots & \alpha_{2n} \\ & & \cdots & \\ \alpha_{m1} & \alpha_{m2} & \cdots & \alpha_{mn} \end{pmatrix} \quad BAC: \begin{pmatrix} \beta_{11} & \beta_{12} & \cdots & \beta_{1n} \\ \beta_{21} & \beta_{22} & \cdots & \beta_{2n} \\ & & \cdots & \\ \beta_{m1} & \beta_{m2} & \cdots & \beta_{mn} \end{pmatrix}$$

图 22-3 行为对抗预案

图 22-3 描述了两方军指挥官的行为对抗预案矩阵,矩阵中的每一行都是前面谈到的各种服务的记录,即行为踪迹。对抗时,双方的指挥官选择使用这些方案进行对抗。

22.5 行为对抗的态势评估

下面,我们介绍一下在网络行为对抗中的红/蓝演习测试的问

题。网络行为对抗必须在规定的环境中实行网络对抗的演习，以了解网络行为对抗预案的可行性、对抗能力和对抗效果。另外，红/蓝演习测试还可以作为国家要害基础设施信息系统承受 **IV** 攻击的能力的评估手段。这种测试与通常的信息安全技术测评认证（CC）的出发点不同，它考查的是国家基础设施信息系统承受信息战的能力，而 CC 测评认证考查的是信息系统承受一般性网络攻击或犯罪的能力。行为对抗能力评估主要是通过红/蓝对抗演习来完成的。这种红/蓝对抗演习是提高行为对抗能力的主要办法。

红/蓝演习测试的示意结构图如图 22-4 所示。

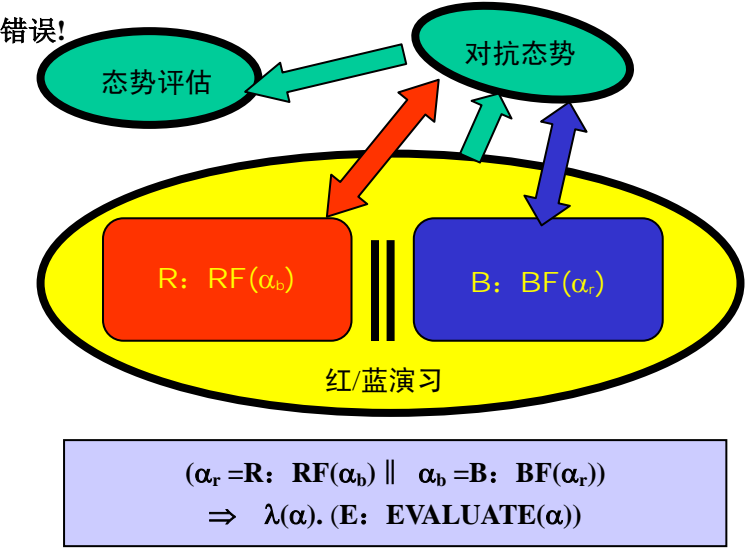


图 22-4 行为对抗红蓝评估

图 22-4 中的 $\alpha_r = R: RF(\alpha_b)$ 是红方的对抗行为，而 $\alpha_b = B: BF(\alpha_r)$ 是蓝方的对抗行为。

第 23 章

群体计算理论

在讨论代理化和代理技术的发展时，曾经谈到可生长和可移动的代理群体体系结构是多代理技术追求的目标。在介绍可计算和算法理论时，谈到的主要是确定计算的计算机模型，包括单机计算模型、多 CPU 或多计算机的并行计算模型以及网络环境中的多计算机分布式计算模型。其中，单机计算模型、多 CPU 或多计算机的并行计算模型，目前可以达到几千个 CPU 和计算机的并行处理能力。对于分布式计算，有以下两种模型：第一种是“网络计算机”模型。计算机网络主要用于解决不同站点计算机之间的通信、资源共享和数据交换。网络计算机把一个网络看成一台计算机进行超级计算。网络计算机把多台计算机通过网络连接成一台计算机进行并行和分布式计算，也称为并行虚拟机（PVM）分布式计算平台模型。第二种是基于中间件技术的 CORBA 分布式计算模型。CORBA（Common Object Request Broker Architecture，公共对象请求代理体系结构）是 OMG 提出的厂商独立、操作系统独立、语言独立、网络基础设施独立的信息平台互操作性规范和标准，其核心是对象请求代理（ORB）部件。ORB 是一种机制，利用这个机制，其中的对象可以彼此相互发出请求和接收响应，并且把这种机制跨越在网络的计算机之间。CORBA 是一个集成广泛对象系统的结构，通过它可以实现网络计算结点连接。不管计算机网络的拓扑结构是什么类型，中间件运行结点都是网络的逻辑中心，扭转网络计算的请求与响应之间的相互关系。

现代计算机作为单机也好，多机也好，其资源和处理能力是十分有限的。从提高计算资源和存储资源来说，过去曾经有多种方法，例如，提高单一处理机速度（可以回忆 CPU 速度不断翻番的发展过程）、采用多处理机并行处理（几十到几千的 CPU 处理能力）、采用分布式计算，等等。从理论上讲，这些计算模型并

没有根本的改变。例如，在计算理论中，单带图灵机与多带图灵机在可计算概念上是等价的。

23.1 群体计算将改变计算理论的面貌

但是，在这里谈到的最重要的计算概念是群体计算，在这种群体计算模型中会产生上述经典模型中所没有的新能力。群体软件的计算或处理能力需要无限的资源。所谓群体计算模型，是指网格（群体）计算，是一种新型计算模型。这种模型具有多代理群体的有组织、可生长和可移动等基本特点，产生了无限资源和处理能力、准不确定计算能力、高智能的分析能力、系统范围的预警能力、主动服务能力和极强的生存能力，下面具体进行分析。

1. 时空统一环境下的无限计算资源

现代计算机模型是非时间概念的，主要是在空间概念中进行讨论，而把时间作为计算的特性（例如时间复杂性）。可以采用多的空间（空间复杂性）来提高速度，减少计算时间。从上述模型定义中可以看出，空间是资源。也就是说，计算机的处理对象是资源，而 CPU 通常不被作为资源对待。但是，在实际中，计算机的处理速度（或者说时间）显然也应当属于一种资源，也应当被调动和配置。在现代计算机资源概念中，即便把空间与时间都看成资源，但通常也是有限的资源，而不可能成为无限资源。但是，在群体计算和群体软件的科学理念中，这种无限资源是可能和现实的。这种无限计算资源的概念将会使现代计算机的体系结构产生革命性的变化。

首先，这种无限资源不是静态空间范围内的概念，而是时空

转换的动态范围内的概念。它认为充分大的空间资源在无限的时间范围内可能变为无限资源,例如,充分大的有限空间资源在无限时间内重复使用,就可以被认为是无限资源。之所以要指明充分大的空间资源,是因为比较小的或不够充分的空间资源不能支持必要的时空转换,或者说没有可用的空间资源在期望的时间范围内进行重复使用。这种无限资源的供应能力不能理解为计算机的无限大存储器和图灵机的无限带长。应当把时空统一环境下的无限计算资源理解为各种类型的计算和资源代理。

于是,可以有如下的假说:

群体计算无限资源假说 对于规定的计算对象,在时空统一环境下应当存在着一个充分大的数,使得其计算资源的供应能力被看做是无限的。

2. 可生长、可移动的群体软件体系

首先,群体软件的概念通常已经远远超过几十、几百、几千甚至几万这样的小数概念,而是一个大数概念,例如几百万、几千万、几亿、几十亿、几百亿、几千亿,甚至更加巨大数量的多代理系统。其次,群体软件的结构不是静态的,而是可生长和可移动的。这种可生长性表示群体中的软件个体是可以自动复制或克隆的,以便扩大其群体规模以适合计算规模和时间需要。可以把这种具有可生长能力的群体资源理解为一种无限的计算资源。所谓可移动,实际上是一种资源的调度和利用概念。人们通常认为调度资源的是一种“第三方”力量,但实际上移动在自然法则中是最好的调度方法。

可生长、可移动群体计算无限资源假说 如果每一个计算代理的计算时间和使用空间都是有限的,在时空统一环境下应当存在着一个充分大的数,使得可生长、可移动的群体软件体系具有

无限计算能力。

下面，介绍可生长、可移动的群体软件体系的不确定计算能力。

说起确定计算机和不确定计算机，大家一定会想起 NP 问题。NP 问题的严格定义是：在多项式时间界限内，确定图灵机与不确定图灵机接受的语言类是否相同。

确定计算机模型在计算机科学中有定义，我们在此不介绍其定义，而是采用比较通俗的方法介绍其概念。凡是编写过程序算法的人都知道，在电子计算机中，不管采用什么算法，其基本的算法概念都是枚举在碰到几种选择时，采用试探方法，即在几种可供选择的路径上，先选择一条进行尝试；如果尝试不成功，便回退到进行选择的地方，再选择第二种可能性；直到做完全部的选择，才可能得出最后的结论（YES 或 NO）。对于一个树形结构的计算问题，其时间复杂性为 2^n ，当 n 较大时，这种计算方法是不可行的。

什么是不确定计算机模型呢？假定有这样一种计算机，当计算中碰到两种以上的选择时，不是像确定计算机那样去进行试探，而是将一台计算机扩展为可供选择的数量那么多的计算机，这些新产生的计算机全部继承原计算机的背景、状态，将所有选择方案同时进行计算，如此不断地执行下去。这种可以随着计算选择要求不断产生新计算机的模型，在人类世界中，只有生物可能做到。即便生物计算机可以做到不确定计算，也不可能是无限的，而是有限不确定计算，但是，如果这种数量非常巨大，甚至比计算的目标数量还大，那么就可以认为是充分的。在单一计算机作为计算部件的时代，半导体、金属、磁性材料和各种有机或无机材料做成的电子计算机作为不确定计算机模型被认为是不可能的。随着网络的广泛应用，在代理技术，尤其是多代理技术、移动代理技术和代理克隆技术得到应用的今天，在一个较大范围或

大范围的网络环境中，这种不确定计算似乎变得可能了。

空间与时间复杂性问题的在群体软件体系面前变得简单了。请看如下的命题：

群体不确定计算命题 可生长、可移动的群体软件体系具有不确定计算能力，可以将时间或空间复杂性从指数型的 2^n 问题转变为低阶多项式问题。

对于确定计算模型计算不确定问题，如果其时间或空间复杂性是指数型的 2^n ，即假定其计算模型在一个二叉树上。我们假定在不确定计算模型中， t_n 表示网络传输时间， t_a 表示代理计算时间， t_c 表示代理克隆时间，完成其计算任务所需要的时间为 $n \cdot (t_n + t_a + t_c)$ 。研究上述问题的目标是针对给定的 n ，期望在不确定计算模型上利用的时间小于在确定计算模型上所利用的时间，即期望 $n \cdot (t_n + t_a + t_c) < 2n \cdot t_a$ 。当然，这是个极端的例子，实际上可以认为能够达到比较低阶的多项式复杂性。所以，我们可以说，可生长、可移动的群体软件体系是一个具有不确定计算能力的新型计算模型。

3. 群体计算是未来的计算概念

群体计算概念是面向可生长、可移动的群体软件的计算概念。对于现代计算机，要达到群体计算能力，可以考虑在近乎无限范围的网络中去实现。在网络时代，尤其是在大范围的网络环境内，处理海量数据和对象以及拥有海量的资源已经成为可能。

23.2 群体计算的行为分析器

笔者在《软件行为学》一书中谈到了群体计算的行为分析器。行为分析对于各种行为模式都是最基本的方法，它可以是对单一

主体行为踪迹的分析，也可以是对多主体多行为的分析。我们把承担行为分析任务的设备称为分析器。行为分析器的分类原则可以规定在多主体多行为的分析之上；对于多主体，可以划分为多主体相关和多主体无关。所谓多主体相关，是表示对于多个主体的行为，需要进行主体相关性分析；所谓多主体无关，是表示对于多个主体的行为，不需要进行主体相关性分析，也就是仅研究单主体的行为相关性问题。多主体无关的多行为分析又可以划分为：单主体前后行为无关分析、单主体有限前后行为相关分析和单主体无限前后行为相关分析；多主体相关的多行为分析同样可以划分为：多主体相关前后行为无关分析、多主体相关有限前后行为相关分析和多主体相关无限前后行为相关分析。行为分析器大致可以划分为如下的类别：

- ◆ 多主体无关多行为分析器 (MultiActions_Free Analyzer)
 - 单主体前后行为无关分析器 (Trace_Free Analyzer)
 - 单主体有限前后行为相关分析器 (Trace_Sensitive Analyzer)
 - 单主体无限前后行为相关分析器 (Infinite Trace Analyzer)
- ◆ 多主体相关多行为分析器 (MultiActions_Sensitive Analyzer)
 - 多主体相关前后行为无关分析器 (MultiActions_Free Analyzer)
 - 多主体相关有限前后行为相关分析器 (MultiActions_Sensitive Analyzer)
 - 多主体相关无限前后行为相关分析器 (Infinite MultiActions Analyzer)

从大的方面讲，多主体无关分析器的复杂性要小于多主体相关分析器。进一步讲，多行为无关分析器的复杂性小于多行为有限前后行为相关分析器，更小于多行为无限前后行为相关分析器。

从理论上讲，我们还可以研究无限主体相关和无限行为相关分析器（如果把无限也看成一个对象）。

所谓相关性，是指依据某种关系、角色、特性和模式而确立的联系。相关性分析有许多方面，例如，在语言学中，有前后文无关文法和前后文有关文法；在网络中，有网络结点无关分析和网络结点相关分析。对于软件行为学，要特别研究主体的相关性和行为的相关性。

对于主体相关性，可以研究主体关系的相关性和主体角色的相关性：

- ◆ **主体关系的相关性** 通过建立、传递、生成、关闭、撤销主体而产生的相关性。例如，一个主体被另一个主体所创立，一个主体被另一个主体所撤销，一个主体被另一个主体和第三个主体共同建立，一个主体被另一个主体和一个客体共同建立，等等。
- ◆ **主体角色的相关性** 依据主体在群体组织中的角色确立相关性。

对于行为相关性，可以讨论行为模式的相关性和行为特性的相关性：

- ◆ **行为模式的相关性** 一个主体或多个主体的行为之间存在着规定模式的联系。
- ◆ **行为特性的相关性** 一个主体或多个主体的行为之间存在着规定特性的联系。

还可以研究其他相关性。下面，我们以单主体行为踪迹分析为例，来介绍分析器的研究方法。

单主体前后行为无关分析器 (Trace-Free Analyzer)

对群体中的行为进行分析时，单主体前后行为无关分析器不考虑不同主体之间的关系，也不考虑每一个主体行为的相关性，认为一个主体的多行为之间是无关的，或者说是不予考虑的。一个主体的多行为问题称为行为踪迹，前后行为无关就是行为踪迹不敏感或不予考虑，仅仅考虑单一行为产生的影响。

前后行为无关分析器的复杂性比较低。令行为踪迹为 $\alpha_1\alpha_2\alpha_3\cdots\alpha_n$ ，不考虑行为相关性，可以有如下公式：

$$T_1=\{\alpha_1\}$$

$$T_2=\{\alpha_2\}$$

$$T_3=\{\alpha_3\}$$

$$T_4=\{\alpha_4\}$$

$$T_5=\{\alpha_5\}$$

.....

$$T_n=\{\alpha_n\}$$

单主体有限前后行为相关分析器(Trace-Sensitive Analyzer)

对群体中的行为进行分析时，单主体有限前后行为相关分析器不考虑不同主体之间的关系，但是考虑每一个主体行为的相关性，认为一个主体的多行为之间是相关的。一个主体的多行为问题称为行为踪迹，前后行为相关就是行为踪迹敏感。

行为踪迹相关性分析有如下结论：

- ◆ 有限前后行为相关分析是可以停机的。
- ◆ 有限前后行为相关的可信性评价是可以停机的。

前后行为相关分析器的复杂性比较高。令行为踪迹为 $\alpha_1\alpha_2\alpha_3\cdots\alpha_n$ ，当前行为为 α_n ，而行为 $\alpha_1, \alpha_2, \alpha_3, \cdots\alpha_{n-1}$ 都为历史行为。以当前行为为核心讨论相关性，看看有多少行为相关的组合。如果定义“ \cdot ”为行为构成踪迹的连接运算，那么可以有如下公式：

$$\begin{aligned}
 T_1 &= \{\alpha_1\} \\
 T_2 &= \{\alpha_2\} \cup T_1 \cdot \{\alpha_2\} = \{\alpha_2, \alpha_1\alpha_2\} \\
 T_3 &= \{\alpha_3\} \cup (T_1 \cup T_2) \cdot \{\alpha_3\} = \{\alpha_3, \alpha_1\alpha_3, \alpha_2\alpha_3, \alpha_1\alpha_2\alpha_3\} \\
 T_4 &= \{\alpha_4\} \cup (T_1 \cup T_2 \cup T_3) \cdot \{\alpha_4\} = \{\alpha_4, \alpha_1\alpha_4, \alpha_2\alpha_4, \alpha_3\alpha_4, \alpha_1\alpha_2\alpha_4, \\
 &\quad \alpha_1\alpha_3\alpha_4, \alpha_2\alpha_3\alpha_4, \alpha_1\alpha_2\alpha_3\alpha_4\} \\
 T_5 &= \{\alpha_5\} \cup (T_1 \cup T_2 \cup T_3 \cup T_4) \cdot \{\alpha_5\} = \{\alpha_5, \alpha_1\alpha_5, \alpha_2\alpha_5, \alpha_3\alpha_5, \alpha_4\alpha_5, \\
 &\quad \alpha_1\alpha_2\alpha_5, \alpha_1\alpha_3\alpha_5, \alpha_1\alpha_4\alpha_5, \alpha_2\alpha_3\alpha_5, \alpha_2\alpha_4\alpha_5, \alpha_3\alpha_4\alpha_5, \alpha_1\alpha_2\alpha_3\alpha_5, \\
 &\quad \alpha_1\alpha_2\alpha_4\alpha_5, \alpha_1\alpha_3\alpha_4\alpha_5, \alpha_2\alpha_3\alpha_4\alpha_5, \alpha_1\alpha_2\alpha_3\alpha_4\alpha_5\} \\
 &\quad \dots\dots \\
 T_n &= \{\alpha_n\} \cup (T_1 \cup T_2 \cup T_3 \cup T_4 \cup \cdots \cup T_{n-1}) \cdot \{\alpha_n\}
 \end{aligned}$$

T_n 有 2^{n-1} 个集合元素，将 $T_1 \cup T_2 \cup T_3 \cup T_4 \cup \cdots \cup T_{n-1} \cup T_n$ 合并起来有 2^{n-1} 个元素。所以，通常需要使用化简复杂性的特别方法来简化分析算法。解决分析器的指数复杂性问题的方法有：

- ◆ 采用多代理系统的不确定计算能力。
- ◆ 配合行为模式相关性和行为特性相关性简化复杂性分析。

单主体无限前后行为相关分析器（Infinite Trace Analyzer）

从理论上讲，规定多行为分析的长度 n 是经验性的，通常， n 越大，相关性分析越严格，但是花费的时间和空间资源也越大。所以，从纯理论上讲，存在着单主体无限前后行为相关分析器：

- ◆ 无限前后行为相关的可信性判定分析器是不停机的。

- ◆ 可以得出已发生行为的可信性评价。
- ◆ 不能得出未来行为的评价，但可以得出未来行为的预期目标。

第 24 章

人类信息化活动的行为学

24.1 人类信息化行为

研究人类的信息化行为是非常重要的事情，它对于建立人类信息化活动的规范，识别哪些信息化活动是安全的和比较容易避免犯罪的、哪些活动是不安全的和容易发生犯罪的，是必要的。研究人类的信息化行为，对于建立完善的信息系统体系结构和科学地提出信息化的总体要求是非常有益的。

人类世界的行为学研究包括个体和群体的各种行为研究。例如，生物圈行为、感知环境的行为、家庭行为、氏族行为、宗教行为、社会行为、文化行为、政治行为、军事行为、经济行为、生产行为、消费行为、学习行为、科学行为、企业行为、团体行为、国家行为，等等。人类的许多行为是在理性指导下认识环境、改造环境并且改造自己，是可以研究其目的、动机、意念和心理等主观特性的。研究人类世界的行为，还必须研究组织行为学。组织的核心问题是组织行为问题。组织行为可以分为组织协同行为和组织业务行为。组织协同行为包括：联系行为、请示行为、指示行为、报告行为、通报行为、视察行为、汇报行为、会议行为、内部业务行为等。这些行为是维持组织结构的行为，没有这些行为，就不能构成真正的组织关系。我们把这些行为称为“组织的协同行为”或“角色行为”。所谓组织业务行为，是指组织和组织中的成员为了实现业务目的所从事的活动或行为。例如，政治行为、经济行为、商业行为、销售行为、开发行为、设计行为、测试行为、社会行为等，我们称这些行为“组织的业务行为”或“组织的功能行为”。

人类世界的行为概念在人类的语言学中已经给出了清晰的定义与描述。在人类语言学中，行为的主体是主语，行为本身称做

谓语句，行为的对象称做宾语、状语或定语等。语言学中还定义了名词、动词（分为及物动词和非及物动词）、副词、形容词等词法学的概念。这些概念范畴的研究显然不属于行为学。行为学研究中最重要概念是“谁做什么”，具体来说，就是“谁来做”、“谁能做”、“做什么”、“能够做什么”、“怎么做”、“为什么做”、“在什么地方做”、“在什么时间做”、“做的结果是什么”、“谁不能做什么”、“不能由谁做”、“必须做什么”、“谁曾经做什么”、“谁将要做什么”，等等，在人类的自然语言中都有体系化的完整描述与说明。其中，“谁做什么”是行为学中最本质的问题，而“能够做什么”、“怎么做”、“为什么做”、“在什么地方做”、“在什么时间做”、“做的结果是什么”、“谁不能做什么”、“不能由谁做”、“必须做什么”等恰恰是对“谁做什么”的进一步的规则性描述和说明。

如果把上述的问题划分为行为的本体语义、行为逻辑语义（行为的规则语义）和行为态势演化语义（行为的动机语义），那么可以得到如表 24-1 的分类。

表 24-1 行为的本体语义、规则语义和动机语义

行为的本体语义	行为的规则语义	行为的动机语义
“谁做什么”	“怎么做”、“在什么地方做”、“在什么时间做”、“谁曾经做什么”、“谁将要做什么”等	“为什么做”、“能够做什么”、“谁不能做什么”、“必须做什么”等

这样的行为描述划分，对于行为结构描述是十分必要的。

一个行为与另一个行为的区别在于行为的三个要素：主体、客体和活动；其中，主体和活动是最基本的要素，而行为的客体

可以有，也可以没有，对于人类活动来说并不是必需的。我们对行为进行一些最基础的划分：非及物行为（无行为客体）和及物行为（有行为客体）。及物行为又可以分为不确定客体的及物行为（简称为不确定及物行为）和确定客体的及物行为（简称为确定及物行为），如图 24-1 所示。行为中有任何一个要素不同，就是不同的行为。例如，“我打开门”与“张三打开门”是不同的行为；“我打开书”与“我打开收音机”是不同的行为；“我拿棍子打狗”与“我用肉喂狗”也是不同的行为。

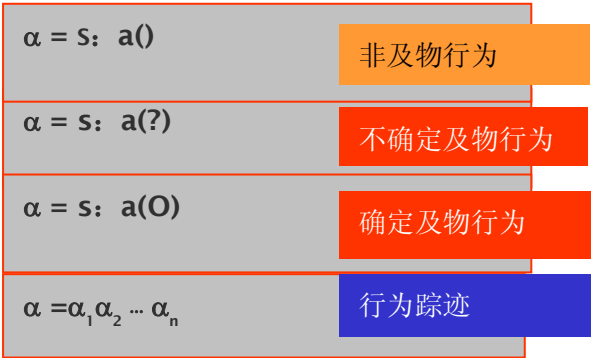


图 24-1 人类世界的行为模式

我们模仿软件行为学中的做法，也对人类信息化行为进行行为的语义研究。经过归纳，人类信息化行为的语义描述方法必须满足如下要求：

行为信息的综合要求

研究行为的主体、行为本身和行为结果是最基本的。作为一个单一行为，有行为过程（活动体）、行为输入、行为输出、行为状态、行为属性、行为结果、行为规则等行为的基本概念。但是，研究行为的信息，不能仅仅局限于行为输入/输出的信息内容

以及输入/输出的满足性，而必须研究行为的结构、模式、分类、环境以及平台等方面的所有可能信息。同时，我们不仅要研究个体的行为，而且要研究群体的有组织行为，包括多主体、多行为和多客体之间的关系和相关性的信息。因此，必须提出全面研究行为信息的综合要求。

行为特性的能力要求

对于人类世界中的人类行为，可以研究与分析其目的性、意图性、必然性和可行性（可管理性和可操作性等）等主观方面的特性以及可理解性、可学习性、可感知性、可适应性、可组织性、可协同性、可对抗性、可信性、有效性、完整性、保密性和连续性等客观方面的特性。但是，在网络世界中研究行为学，主要讨论的是可计算性、可理解性、可代理性、可自治性、可信性、有效性、完整性、保密性和连续性等客观特性，要求对个体和群体的行为以及行为的结果进行研究，对内容的真实性、可信性、保密性和完整性等研究具有支持能力。

行为模式的应用要求

所谓行为模式，是指行为所属的应用模式，包括组织行为、业务行为、任务行为、管理行为、商业行为、渠道行为、产品服务行为、开发行为、设计行为和测试行为等。行为模式要求研究这些领域中行为的领域范畴的特性、环境和属性等概念，还要求研究群体行为基础性的模式，例如行为协同模式、伴侣行为模式、行为服务模式、行为管理模式、行为控制模式、行为监管模式、行为认证模式、行为对抗模式和行为博弈模式等。

行为平台的系统要求

这种语义描述方法必须在全局意义上对系统设计提供一体化的支持，不仅要支持群体中规范代理的系统结构设计，而且还要

支持不同层次群体意义上的系统结构设计，同时提供统一方式的软件工程支持。

信息化建设将人类世界中的许多行为信息化了。所谓信息化行为，就是人类使用信息系统或通过信息系统行为完成行为目标。实现行为信息化的媒体有：电话、电报、传真、电子邮件、文件、打印设施、电话会议系统、电视会议系统、网络会话系统、办公信息系统和业务应用系统等。目前，银行、证券、保险、交通、民航、电力、电信、文化和人民生活等数以千计或万计的服务都实现了信息化，其行为是信息化行为。人类使用信息系统的行为是整个人类行为集合的子集，是其中的一部分，而且这一部分越来越大，所发挥的作用也越来越大。对于人类信息系统的行为，客体有物理形式、电子形式和数字形式等几种。

但是，人类的信息化行为已经超越了这种直觉概念，例如，信息系统（软件/硬件）设计、开发、测试、实现、生产、维护、培训、评估、技术法规制定、标准制定、非法入侵、网络犯罪、网络执法和打击网络犯罪等都属于人类信息化行为。

24.2 可信息化条件讨论

下面，我们讨论本节的最后一个问题：什么是可信息化的行为。在上面列举的人类行为中，人类生物圈行为是不能信息化的，例如，吃饭、睡觉、喝水等都是人类生物圈行为，不是可信息化行为。但是，感知环境的行为、社会行为、文化行为、政治行为、军事行为、经济行为、生产行为、消费行为、学习行为、科学行为中大部分是可信息化的。一些组织中的协同行为和业务行为，例如联系行为、请示行为、指示行为、报告行为、通报行为、视

察行为、汇报行为、会议行为等，也是可信息化的。通常，与视觉和听觉相关或者可转换成与视觉和听觉相关的行为，都是可信息化的。我们为可信息化制定了如下四个层次的标准：

- ◆ **行为信息能够数字化** 首先，行为信息能够信号化。例如，压力数据、温度数据、密度、成分数据等都可以用仪器和表格方式显示。其次，这种信号可以被数字化。即便是人类感觉（触觉、味觉和嗅觉），也可以数字化。也就是说，与行为相关的信息是可以作为数字信息传输、存储和处理的，行为是可信息化的。
- ◆ **行为信息能够可视化** 行为信息可信号化和数字化，但人们却不一定能够理解这些信号与数字信息的含义。为了理解信号与数字信息的含义，可以通过可视化进行理解。例如，环境信息能变成可视化的信息（例如，压力、温度、密度和成分都可以可视化）。人类的触觉、味觉和嗅觉信息可变成信号，这些信号与信息不能恢复成人类的触觉、味觉和嗅觉，而可以恢复成一种可视化的信息。
- ◆ **行为可代理化** 行为可以委托其他主体执行称做代理化。这种代理可以是物质形式的设备（机器人），也可以是网络世界中的虚拟主体。代理可以是以个体形式存在的，也可以是以有组织群体形式存在的。人类应保留一些不可以或不能够代理化的行为。
- ◆ **行为可智能化** 行为是依据知识、逻辑和模式等规则理性实施的。

综上所述，我们把行为信息能够数字化、可视化以及行为可代理化、智能化作为可信息化的评价依据。

第 25 章

信息化总体学与体系结构

25.1 体系结构研究实践及存在的问题

信息化体系结构的研究已经有了相当的基础，在标准化和实践方面都有一定的成果。下面，首先来看看国际标准化组织对体系结构的研究情况。

国际标准化组织对信息化体系结构给予了极大的关注。体系结构是信息化总体设计中最重要概念。信息化总体设计同样需要采用比较规范和标准化的方法，笔者建议采用以 IEEE 610.12 为开创标志的体系结构方法（1990）。后来，这种方法又被 IEEE STD 1471 标准（2000）和 ISO/IEC 开放分布式处理-引用模式（Open Distributed Processing-Reference Model）及其开放体系结构框架（The Open Group Architecture Framework, TOGAF）发扬光大。IEEE 组织在 IEEE STD 1471（2000）中修改了体系结构概念，将其定义为：一个体系结构是包含在系统中部件基本组织、相互关系与它们同环境的关系和指导设计与评估原则的结构体系。这个体系结构定义被运用在美国国防部的体系结构框架（DoDAF）中。

另外，OMG 对软件体系结构也进行了深入研究，其中最著名的是统一建模语言（Unified Modeling Language, UML）和模式驱动的体系结构（Model Driven Architecture, DMA）。

下面，再来看看美国国防部对信息化体系结构的研究情况。首先，在实践方面，这种体系结构方法应用于美国军方的 DII COE 计划实践中，1996 年产生了 C4ISR 体系结构框架第 1 版（C4ISR Architecture Framework V1.0），1997 年产生了 C4ISR 体系结构框架第 2 版（C4ISR Architecture Framework V2.0）。在这以后，美国国防部将 C4ISR 体系结构框架第 2 版重新命名为美国国防部体系结构第 1 版（DoD Architecture Framework, Version 1.0），2003

年 8 月正式发布，作为在整个国防部范围内信息化总体设计的标准化方法。在这个体系结构文件中定义了 2 个全视图产品（AV- 1, AV-2）、7 类运营视图产品（OV- 1, OV-2, OV-3, OV-4, OV-5, OV-6a, OV-6b, OV-6c, OV-7）、11 类系统视图产品（SV- 1, SV-2, SV-3, SV-4, SV-5, SV-6, SV-7, SV-8, SV-9, SV- 10a, SV- 10b, SV- 10c, SV- 11）和 2 个技术视图产品（TV-1, TV-2），如表 25- 1所示。利用这些体系结构产品，已经产生了 DII COE、SHADE（数据共享工程）、LISI（Levels of Information System Interoperability）、JTA（联合技术体系结构）、JOA（联合运营体系结构）、JSA（联合系统体系结构）、CADM 和 UJTL 等成熟的综合体系结构产品系列。与此同时，美国国防部考虑到全球化需要，又在上述研究的基础上，进一步提出了 GIG 体系结构的概念和标准化方法。我们需要明白，美国国防部的 DoDAF 体系结构已经与 IEEE 1471有了根本的不同，它有运营体系结构概念，而不仅仅是业务体系结构、技术体系结构、数据/信息体系结构和应用体系结构；它承认主体的结构性，研究主体的运营体系结构，而不仅仅考虑系统的体系结构。因此，DoDAF 体系结构标准得到的评价很高。

表 25-1 DoDAF 中定义的视图产品

应用视图	框架产品	框架产品名称	说明
全视图	AV-1	概述	作用、范围、目的、用户、环境等描述与分析
	AV-2	综合字典	在所有产品中定义的具有所有术语解释的体系结构
运营视图	OV-1	高级运营概念描述	对运营语言的文字和图表等方式的描述

(续表)

应用视图	框架产品	框架产品名称	说明
运营视图	OV-2	运营结点连接性描述	运营结点及其连接性
	OV-3	运营信息交换表	运营中交换的信息
	OV-4	组织关系图表	组织结构、职能、角色和其他组织关系
	OV-5	运营活动模式	描述运营活动、能力、活动之间的关系、输入和输出
	OV-6a	运营规则模式	描述运营活动识别业务规则和约束
	OV-6b	运营状态转移	描述运营活动识别业务过程
	OV-6c	运营事件描述	描述运营活动事件序列和设定方案
	OV-7	逻辑数据模式	运营语言的数据指标体系
系统视图	SV-1	系统接口描述	描述在不同层面的系统识别、接口等信息
	SV-2	系统通信描述	描述不同层面的系统结点之间的通信
	SV-3	系统-系统表格	描述给定体系结构中系统之间的关系
	SV-4	系统功能性描述	描述系统功能执行和系统功能之间的数据流
	SV-5	运营活动与系统功能的可追踪性表格	描述系统能力或系统功能与运营活动的关系

(续表)

应用视图	框架产品	框架产品名称	说明
系统视图	SV-6	系统数据交换表格	提供系统之间的数据交换元素和交换属性
	SV-7	系统执行特性表格	描述系统视图元素的执行特性
	SV-8	系统演化描述	系统不断完善和发展的阶段性描述
	SV-9	系统技术前瞻性描述	描述对未来体系结构发展产生影响的新技术和软硬件
	SV-10a	系统规则描述	系统设计与实现的约束条件集合
	SV-10b	系统状态转移描述	描述系统实践的反应
	SV-10c	系统事件踪迹描述	在运营视图中描述的事件序列在系统执行中的情况描述
	SV-11	物理框架	逻辑数据模式的物理实现
技术视图	TV-1	技术标准框架	列出给定体系结构系统视图元素所采用的标准清单
	TV-2	技术标准前瞻性	描述当前系统视图元素新显现的标准和影响

在信息化体系结构的研究方面，美国电子政务也有自己的成果。美国政府电子政务的体系结构称为联邦企业体系结构(Federal Enterprise Architecture，简称 FEA)，是由美国政府的管理与预算办公室(Office of Management and Budget，OMB)提出的。这个总体体系结构方法虽然借鉴了美军的体系结构方法，但是具有一般政府信息化的特点。它主要是站在美国总统的角度，以公众

服务为中心来看美国电子政务的体系结构的（即不是美国政府信息化的全部）。从本世纪初开始，通过一段时期的不断研讨和修改，在 2003 年到 2004 年之间先后提出了 FEA V 1.0 的所有文件。这个体系结构的标准化方法提出了业务引用模式（Business Reference Model, BRM, 美国政府全局电子政务的业务模型）V2.0、执行引用模式（Performance Reference Model, PRM）V 1.0、服务引用模式（Service Reference Model, SRM）V 1.0、数据引用模式（Data Reference Model, DRM）V 1.0 和技术引用模式（Technical Reference Model, TRM）V 1.1 等体系结构模式范畴。FEA 的体系结构框架概念借鉴了 TOGAF 的体系结构分类，同样定义了业务体系结构（Business Architecture）、数据体系结构（Data Architecture）、应用体系结构（Application Architecture）和技术体系结构（Technology Architecture）。美国 FEA PMO 对于在 2005 年及以后的年份如何加强信息化体系结构标准化方法的研究与应用同样做出了规划，制定了行动指南。显然，这种体系结构标准化方法还需要进一步完善，例如，要将这种方法运用到各领域中去，而不能仅仅停留在从白宫角度看电子政务。

上述体系结构方法对于中国的信息化体系结构设计具有很好的借鉴作用。在中国的信息化实践中，通过尝试这些体系结构的标准化方法，我们有了自己的认识与体会，这种体会尤其表现在中国信息化建设的特点中，我们逐步找到了适合中国国情的体系结构标准化方法。上述体系结构标准化方法没有考虑一些新技术的发展和体系结构带来的变化，对多代理技术发展带来的群体计算、群体软件新型理念没有研究，缺乏认识（或者说没有达成共识，因为标准化通常晚于技术发展趋势）。

从上述关于信息化体系结构的讨论中可以看出，体系结构的标准化方法是缺少理论研究的，有如下的问题需要注意：

- ◆ IEEE 1471和 TOGAF 的系统概念过于任意。
- ◆ 应当区分关注者在系统之内和在系统之外。
- ◆ 系统之内的关注者本身也是有体系结构的。
- ◆ 信息化运营、管理和服务模式在不断变化,业务与技术组织体系结构也在不断变化。
- ◆ 信息系统的体系结构也发生了很大变化。
- ◆ 关注点认识错了,体系结构便失去了基础。
- ◆ 关注点理论研究是体系结构研究的基础。

体系结构的研究与标准的建设必须建立在坚实的理论研究基础之上。这个用来指导体系结构研究的理论实际上是人类世界(人类活动的世界)和网络世界(网络系统中主体活动的世界)两个行为学理论体系。

研究人类信息化行为的目的是研究人类信息化行为的体系结构概念。信息化体系结构研究主要是研究人类世界中主体如何使用信息化的行为,这些行为包括维系组织关系的“组织协同行为”和体现组织存在价值的“组织业务行为”,这两种行为汇合在一起构成运营行为。研究组织运营行为的结构关系描述,称为运营体系结构。针对运营体系结构要求进行信息化的建设,建立一些信息系统,这些系统在网络世界中同样通过行为来表现系统的功能和能力,这样的系统描述称为系统体系结构。系统体系结构与运营体系结构的一致性是通过系统行为和运营行为之间的检查达到的。

在信息化体系结构或总体学的研究中,说明问题和事物最重本质的核心概念是定义与引用。在人类世界中,例如,科学成果的第一次发布可以理解为定义,而考查其成果价值最基本的指标是被引用的描述。在运营的范畴内,人类活动的连接关系也是一

种在规定部门和人员之间的输入与输出或者信息的交换，从宏观看属于一种定义与引用关系。在系统范畴内，系统结点的连接关系通常是通信或信息交换的接口或者 API 的连接，从说明的角度看也是一种定义与引用关系。在软件领域中，程序连接也是通过定义与引用关系建立起来的，软件是通过定义建立入口点，期望其他软件进入和调用。在技术范畴内的标准化结构，技术标准都有依赖的基础（引用其他）和被其他技术标准进一步引用的关系。于是，我们得出结论：体系结构采用定义与引用的描述方法是基本概念。

现代系统中不仅引入了网络概念，而且还引入了计算机系统和人的概念。现代系统模型可以定义许多种形式，其中一种形式是如下的类型表达式所描述的内容：

```
System = Computers----- 计算机集合
        × Networks----- 网络系统
        × Human----- 用户或管理者
Computers = (Computer)N
Computer = Computer_Ide----- 计算机标识
        × Hardware ----- 硬件
        × BIOS ----- 基本输入/输出系统
        × OS----- 操作系统
        × AppSoftwares----- 应用软件
        × HCI----- 人机接口
OS = (Subject)+ × (Software)+----- 操作系统主体与软件
Subject = Process + Agent + MultiAgent + ...
```

网络概念用数学的方法可以描述为如下表达式：

```
Networks = {(c1,c2) | c1,c2 :
Computer and λp•transport(p,c1,c2) is enable and p :
Package,transport Package×Computer×Computer×Network
s →Networks}
```


两个世界的主体完成的。网络世界中的主体可以是系统进程、用户进程、代理和多代理。

25.2 体系结构理论研究

基于上述考虑，我们应当讨论的是一种新型的结构概念：主体结构、行为结构、客体结构和规则结构（见图 25-1）。

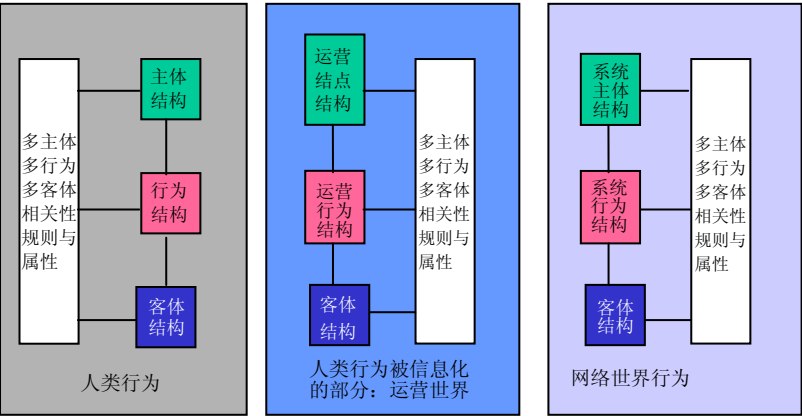


图 25-1 人类行为、信息化行为和网络世界行为结构

主体结构

主体结构是定义人类群体的一种结构。从其定义中可以看出，它是依据某种规则通过行为实践表达行为关系的结构。一个组织中两个人的关系不是通过排座位的椅子的关系来表达的，也不是通过任命书建立的关系来表达的，而是通过人之间的行为关系来表达的，例如，领导行为和被领导行为。也就是说，如果没有行为关系的表达，那么组织关系是没有意义的。企业中许多组织结构的关系都是依据行为关系或行为结构建立的。这种行为可以是

内部组织、管理行为，也可以是组织内部或外部的业务行为、服务行为或渠道行为。也就是说，主体结点的连线所表示的是结点的行为关系，是一种“活动线”、“行为线”、“需求线”、“管理线”或“服务线”。

描述主体结构时，如果在主体结点的连线上标明所有的“活动线”，那么这个主体结构关系将过于复杂与凌乱。因此，主体结构至少应当根据组织群体的行为分类（例如，组织管理行为、业务行为和其他行为）来分别描述主体结构关系。

主体结构关系可以是逻辑关系，也可以是虚拟关系或物理关系。主体结构同样具有时间特性和某种动态特性（或阶段特性）。我们可以把主体结构描述为逻辑结构（运营说明结构）、虚拟结构（设计开发的说明结构）和物理存在的结构。但是，在研究人类活动时，讨论的主要是逻辑结构。

主体结构具有分层或分级（主体组织根据主体内部的业务划分成一些部门，这些部门属于主体的组成部分或者为下级单位）、面向空间分布（主体组织分布到不同的地理位置和地区上，例如分布在不同的国家、地区和省市）和面向业务分类（例如政府机构、军事机构、学校机构、咨询机构、研究机构、设计机构、运营商结构、产品供应商结构、服务供应商结构、销售机构）等特性。从结构特点看，主体结构具有“条块”结构特性，例如，信息化管理的组织体系通常是依据业务划分“条块”的，构成纵向的脉络和链条。这种业务链条是与企业或机构开展的业务类型数量相关的。

一个信息化体系结构实际上是非常丰富的，对于不同的关注者（信息化体系的拥有者、运营者、规划者、设计者、开发者、执行者、管理者、维护者、服务者和支持者等），需要建立的信息化运营方式和体系结构不同，需要建立的信息系统体系结构也

不同。例如，运营者、管理者主要关注运营或管理模式，即运营体系结构或管理体系结构；而规划者、设计者和开发者主要关注设计什么样的系统能够实现运营模式要求的内容；执行者、维护者、服务者和支持者主要关注选择什么样的技术体制或者技术体系结构来实现系统体系结构中规定的组成部分；第三方评估体系结构的关注者关注的是这些体系结构的价值、性能、能力和效果等方面的内容。

运营体系结构的主体是指信息化体系的拥有者、运营者、管理者、使用者、规划者和设计者等，其中，设计者是既关心运营体系结构又关心系统体系结构的人员。运营体系结构的关注者通常不包括系统的开发者、实现者、测试者和维修者等，因为这些人员通常没有必要关注运营体系结构。

运营主体结点与主体组织结点不同。例如，信息化运营体制的业务运营中心、数据中心、信息交换中心、票据/文档交换中心、认证中心、公众服务中心、通信中心、风险监管中心、应急中心、管理中心、配置网络服务中心、软件活动支持中心、端结点用户等对于信息化来说就是运营结点。对于企业来说，这个结构关系显然与企业股东会、董事会、总经理、副总经理、财务部、人力资源部、产品部、市场部、销售部、研发部、服务部、办公室等部门的行政管理结构没有关系，它们两个是完全不同的结构。应当说，组织结构中的每一个组织结点和结点中的每一个人都是运营体系结构的用户端结点，他们都是运营结点中信息系统的用户。

主体结构描述方法可以采用多种形式，通常采用树形组织结构图。

行为结构

首先，行为结构是与主体结构相关的，因为所有的行为都是

主体的行为。也就是说，不能脱离主体结构概念来讨论行为结构。主体组织的行为分为组织行为和业务行为，表现在群体组织行为与个体行为之间，例如，企业行为、部门行为和企业员工行为构成了一种结构关系。在讨论主体结构时曾讲过，组织的管理关系是通过成员的行为来建立的，这说明表达组织关系的行为具有与组织管理相同的结构关系。组织的业务关系同样也是以业务行为来建立和维系的，也具有与业务关系相同的结构关系。行为结构有拓扑结构关系，也有层次结构关系。从另一个角度看，这种行为结构是多主体的多行为结构，表现在多个主体行为之间的结构关系上。

其次，行为结构考察一个主体的行为（可以是多个行为），把这些行为按照时间的顺序排成一个序列，便构成了一个多行为踪迹。这些行为踪迹中的行为之间是有关系的，具有流程的特点。行为流程可以描述行为的输入/输出关系和因果关系等。这种因果关系表现在当前的状态和环境中，可以选择的行为是可能的后续行为，一个行为执行后，必然会产生相应的结果，对主体、客体或者环境产生影响。也就是说，一个主体的行为或行为踪迹不仅影响自身和环境，还可能影响其他主体及其环境，改变它们的状态，甚至直接影响其他主体行为序列的形成过程，成为其行为或行为踪迹发生的原因和前提。例如，在军事上，多路的军事行为构成了战场作战的结构关系，当然也包括敌我双方军事行为结构关系。

最后，行为结构具有层次（分级）、拓扑（主体/客体分布逻辑关系）、空间（主体/客体空间关系）和时间（行为顺序）结构关系，同时也具有相应的结构。

综上所述，行为结构定义应当反映出行为具有分层，面向多主体、多行为，面向主体与客体分布和时间有序的特性。

客体结构

人类活动的客体可以是物质存在的一切实体，也可以是主体自身和其他主体。这种客体对象在许多情况下也是存在结构关系的。如果客体本身就是作为行为主体的人或人类组织，那么其结构性是显然的。非人类的实体也具有结构形式，例如，动物、植物和其他一切生物体系具有结构性也是显然的。对于非生物特性的自然物质，由于其物理或化学的性质，也会产生结构性现象，例如地理、气象和天体等。

规则结构

规则之间也有相互关系。规则是主体、行为和客体的规则，由于主体、行为和客体有结构，那么规则也有结构。图 25-2 显示了一个主体结构图。

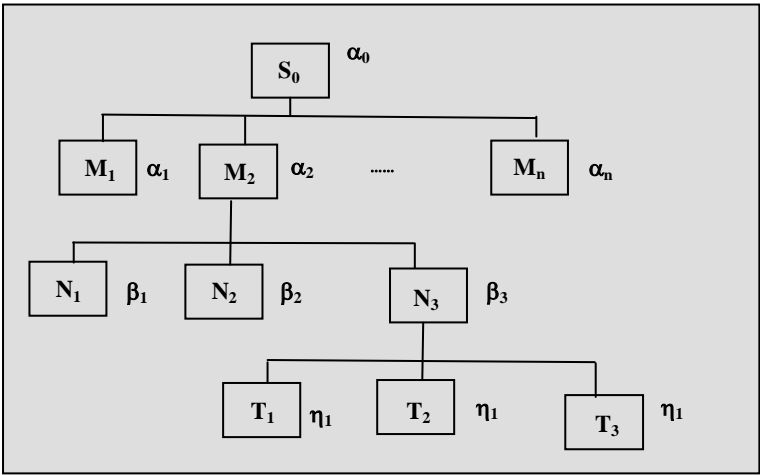


图 25-2 主体结构

为了区别程序设计语言的文法定义方法，在体系结构研究中

表示一个组织机构图, 采用 “::” 定义符号, 针对图 25-2 所示的组织结构图有如下的主体结构定义:

$S_0:: S_1 S_2 S_3 \cdots S_n$, 令 α_i 是主体 S_i 的行为, $i \in \{0, 1, 2, \cdots, n\}$, 并有 $\alpha_0 = S_0: \text{MngtMultiAction}(\alpha_1 \parallel \alpha_2 \parallel \cdots \parallel \alpha_n)$ 。

1. 对于行为 $\alpha_0 = S_0: \text{MngtMultiAction}(\alpha_1 \parallel \alpha_2 \parallel \cdots \parallel \alpha_n)$, 有如下的逻辑描述:

$$\begin{aligned} \alpha_0 &= S_0: \text{MngtMultiAction}(\alpha_1 \parallel \alpha_2 \parallel \cdots \parallel \alpha_n) \\ \alpha_0 &= S_0: \text{Collaborate}(\alpha_{01} \parallel \alpha_{02} \parallel \cdots \parallel \alpha_{0n}) \\ \alpha_{01} &= S_0: \text{MngtOneAction}(\alpha_1) \\ \alpha_{02} &= S_0: \text{MngtOneAction}(\alpha_2) \\ &\cdots \\ \alpha_{0n} &= S_0: \text{MngtOneAction}(\alpha_n) \\ &\cdots \end{aligned}$$

下面, 我们对上述的描述进行一些说明。行为公式 $\alpha_0 = S_0: \text{MngtMultiAction}(\alpha_1 \parallel \alpha_2 \parallel \alpha_3)$ 是很容易理解的, 即管理者对下属的多个人员的行为进行多行为管理。对于多行为管理, 如果需要进一步定义, 那么 $S_0: \text{MngtMultiAction}(\alpha_1 \parallel \alpha_2 \parallel \alpha_3)$ 可以描述成对多个单行为管理的描述, 例如 $S_0: \text{MngtMultiAction}(\alpha_1 \parallel \alpha_2 \parallel \alpha_3) = S_0: \text{MngtOneAction}(\alpha_1) \parallel S_0: \text{MngtOneAction}(\alpha_2) \parallel S_0: \text{MngtOneAction}(\alpha_3)$ 。显然, 这个公式过于简单, 与实际情况不符合。所以, 我们在上面定义了 $S_0: \text{MngtMultiAction}(\alpha_1 \parallel \alpha_2 \parallel \alpha_3) = S_0: \text{Collaborate}(S_0: \text{MngtOneAction}(\alpha_1) \parallel S_0: \text{MngtOneAction}(\alpha_2) \parallel S_0: \text{MngtOneAction}(\alpha_3))$ 的形式, 表示多行为管理是在对每一个单行为管理的基础上, 协调它们之间的相关问题。

对于第二层次的主体行为结构, 可以描述如下:

$M_2: :N_1N_2N_3$, 令 α_2 是主体 M_2 的行为, 再令 $\beta_1, \beta_2, \beta_3$ 分别是主体 N_1, N_2, N_3 的行为, 并有 $\alpha_2= S_2: \text{MngtMultiAction}(\beta_1 \parallel \beta_2 \parallel \beta_3)$ 。

2. 对于行为 $\alpha_2=M_2: \text{MngtMultiAction}(\beta_1 \parallel \beta_2 \parallel \beta_3)$, 有如下的逻辑描述:

$$\begin{aligned}\alpha_2 &= M_2: \text{MngtMultiAction}(\beta_1 \parallel \beta_2 \parallel \beta_3) \\ &= M_2: \text{Collaborate}(\alpha_{21} \parallel \alpha_{22} \parallel \alpha_{23}) \\ \alpha_{21} &= M_2: \text{MngtOneAction}(\beta_1) \\ \alpha_{22} &= M_2: \text{MngtOneAction}(\beta_2) \\ \alpha_{23} &= M_2: \text{MngtOneAction}(\beta_3) \\ &\dots\dots\end{aligned}$$

对于第三层次的主体行为结构, 可以描述如下:

$N_3: :T_1T_2T_3$, 令 β_3 是主体 N_3 的行为, 再令 η_1, η_2, η_3 分别是主体 T_1, T_2, T_3 的行为, 并有 $\beta_3= N_3: \text{MngtMultiAction}(\eta_1 \parallel \eta_2 \parallel \eta_3)$ 。

3. 对于行为 $\beta_3= N_3: \text{MngtMultiAction}(\eta_1 \parallel \eta_2 \parallel \eta_3)$, 有如下的逻辑描述:

$$\begin{aligned}\beta_3 &= N_3: \text{MngtMultiAction}(\eta_1 \parallel \eta_2 \parallel \eta_3) \\ &= N_3: \text{Collaborate}(\beta_{31} \parallel \beta_{32} \parallel \beta_{33}) \\ \beta_{31} &= N_3: \text{MngtOneAction}(\eta_1) \\ \beta_{32} &= N_3: \text{MngtOneAction}(\eta_2) \\ \beta_{33} &= N_3: \text{MngtOneAction}(\eta_3) \\ &\dots\dots\end{aligned}$$

4. 对于终端行为 η_1, η_2, η_3 , 有如下的逻辑描述:

```
η1=T1: open→O  
η2= T2: read(O)  
η3=(T3: send(O))⇒( X: receive(O))
```

需要特别说明的是，平行行为并不必是同时或并发的。在通常意义下，多主体的多行为才有这种平行行为关系。但是，单一主体的多行为不能通过行为顺序关系或行为踪迹表达一种时间序列的多行为关系，同样可以采用平行行为的操作。例如， $S_0: G(S_0: H(\alpha_1) \parallel S_0: H(\alpha_2) \parallel S_0: H(\alpha_3))$ 中的三个独立行为 $S_0: H(\alpha_1)$ ， $S_0: H(\alpha_2)$ ， $S_0: H(\alpha_3)$ 是顺序不确定的多行为，它们有同一个主体。

以往谈到系统体系结构时，不谈系统的物理结构，而主要谈系统的组成、系统的拓扑结构、系统的层次结构、系统接口的连接和系统功能等概念。例如，系统采用如图 25-3 所示的运营结点连接方式。

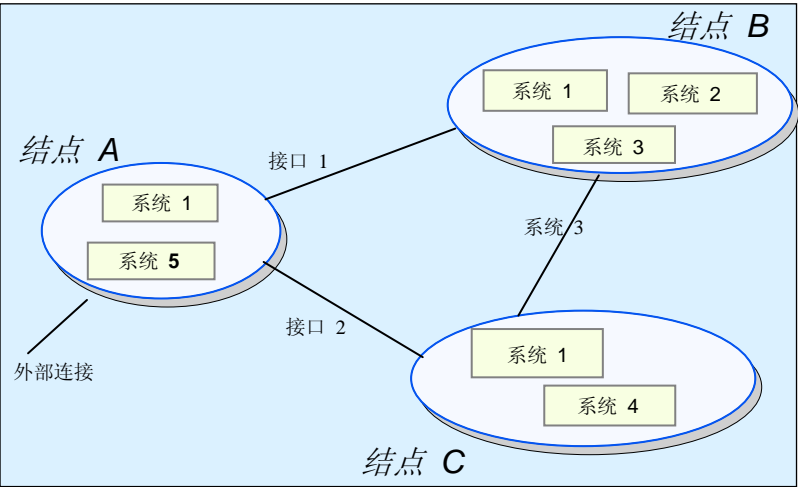


图 25-3 系统结点连接结构

根据定义与引用的关系，可以把计算机中的软件系统划分为应用基础平台、应用平台和应用系统。对外通过通信与网络平台与系统连接，与人的交互通过人机界面和交互系统实现，如图 25-4 所示。

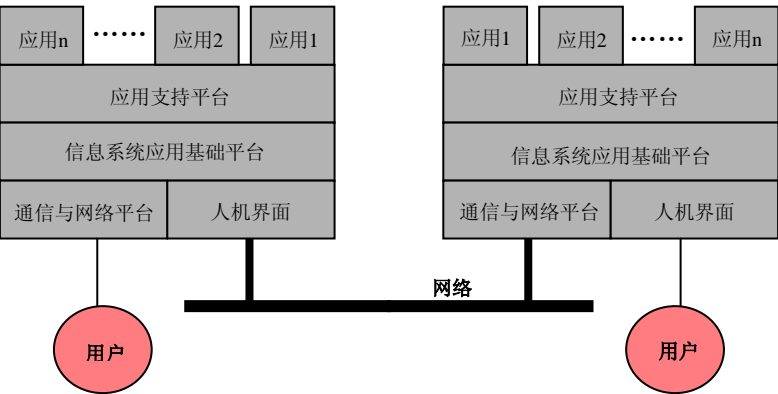


图 25-4 软件系统定义应用层次结构

其实，系统结构不仅仅是软硬件之间的连接关系和定义与引用的关系。需要告诉读者的是，在定义系统体系结构时，第一位的工作是定义系统逻辑体系结构，按照本书的说法是定义网络世界中的主体结构、行为结构、客体结构与规则结构。在建立好了网络世界中的主体结构、行为结构、客体结构和规则结构之后，再来讨论系统的虚拟体系结构。例如，在上述的例子中，网络世界中的主体结构与行为结构如图 25-5 所示。

通过将上述的研究成果结合起来，并使之更加体系化和理论化，我们提出了“三视图+多引用模式标准化方法”。在运营体系结构中扩充为四个引用模式，即：业务组织引用模式（BORM）、业务行为引用模式（BARM）、业务数据体系引用模式（BDRM）

和业务规则引用模式（BRRM），使体系结构的描述更加完善与全面，将 DoDAF 中提出的体系结构产品 OV-1, OV-2, OV-3, OV-4, OV-5, OV-6a, OV-6b, OV-6c, OV-7 并入到上述四大模式中，重新定义为引用模式的产品。对于系统体系结构，本书中定义了系统主体引用模式（SSRM）、系统行为引用模式（SARM）、系统数据引用模式（SDRM）和系统规则引用模式（SRRM），将 DoDAF 中提出的体系结构产品 SV-1, SV-2, SV-3, SV-4, SV-5, SV-6, SV-7, SV-8, SV-9, SV-10a, SV-10b, SV-10c, SV-11 也并入到上述的系统体系结构四大模式中，并对其进行了修改，将其重新定义为引用模式的产品。这四个模式可以与运营体系结构的四个模式建立起一定的对应关系。

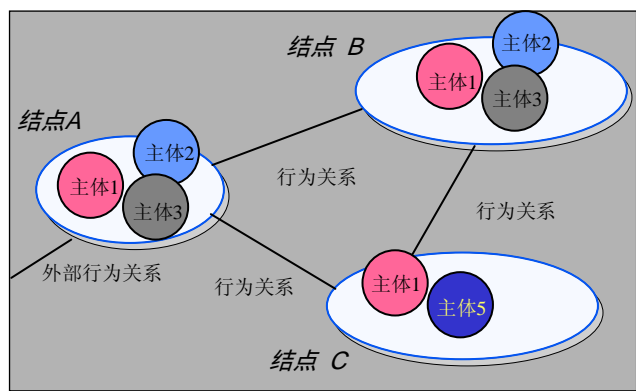
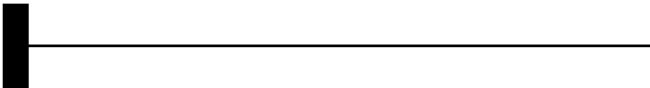


图 25-5 系统主体行为结构

第 26 章

可信网络世界体系结构框架（TCAF）

26.1 可信网络世界的可信概念

首先,我们来介绍一下可信概念。

在美国国防部的 TCSEC 标准中,最早定义了可信计算基 TCB (Trusted Computing Base) 的概念。这种 TCB 安全核的概念在操作系统和建设在其上的数据库管理系统以及应用系统中得到了普遍的采用。在操作系统的安全观念中,按照角色(例如管理员、超级用户和普通用户等)给予引用系统服务的一定权限。在 TCSEC 中提出的可信概念实际上是一种授权概念,即那个时代的可信概念是“授权可信”。这种可信概念只看身份和权限,而不看表现行为。现代的可信概念应当是“既看身份与权限,又看行为表现”,最终落实在行为可信概念上。行为可信建立在行为或多行为或历史行为进行考查(监管、认证和控制)的基础之上。

在这里,对行为的可信性做如下定义:

行为可信性用于考查行为预期性的满足,这种预期性满足是在多主体多行为范畴内,对行为主体、行为客体、行为环境、行为性质、行为输入/输出、行为过程、行为状态、行为属性等在符合必须遵守的要求、约定、规定、规则和法律满足性等方面进行认识与评价。行为的可信性还表现在对行为预期结果和实际结果之间差距的认识、把握、控制、调整 and 改变上。

网络与系统的安全与可信问题是需要认真研究的。所谓可信,是对行为可信性预期的满足,在上面已经详细讨论过了。而安全性最重要的不是考查一个系统是否符合安全技术标准并具有安全功能和服务能力,而是考查这些安全功能和服务能力是否真的发挥作用,系统的安全问题是否被真正解决,为用户建立安全的感知和信心,使之对实际的安全效果与预期的安全效果之间

的差距具有认识，有能力把握、控制和改变这种差距。这种安全概念的认识实际上是通过无数的教训而获得的。对于一个网络、系统或者航天飞机的安全，不能仅仅考虑采用了什么安全防护技术，还应考虑这些安全防护技术是否达到了安全的效果和满足了预期的要求。可信是建立信息化安全效益的基本概念，换句话说，仅仅考虑信息化安全技术服务标准能力的建设，而不重视可信服务能力的建设，这种信息化建设就是不讲安全效益。

建立可信概念，要涉及到可信服务、可信技术、可信系统（软件）和可信用户等概念。

26.2 可信网络世界的平台概念

其次，我们来介绍一下平台概念。

平台是一个经常出现在各种技术文献中的术语。什么是平台呢？平台概念是有针对性的，对于开发而言，平台实际上是一个开发支持环境；对于系统集成而言，平台是一个支持系统之间进行链接（包括静态和动态链接）的体系或环境；对于运行而言，平台是一个支持软件或系统依照一定模式（创立、激活、执行、中断、停止执行、撤销）运行的环境；对于应用来说，平台是支持应用的操作环境；对于管理来说，平台是提供资源分配、优化、管理和撤销等功能的支持环境。所有的平台都是使系统相互结合发挥功能的环境体系。平台本身并没有提供一般意义上的业务功能操作，而是由平台上的系统提供功能服务，平台的作用是把这此系统协调在一起，以便它们更好地发挥作用。

平台的定义如下：

平台是为系统提供互操作性及其他服务的环境。

实现系统之间的互联、互通和互操作有不同的目标，例如，实现综合业务应用，实现多系统之间的信息交换、交易事务综合处理，进行多系统之间的综合管理，实现多系统的综合运营等。

平台通常提供特有的功能服务，例如：

- ◆ 多系统之间的信息共享服务
- ◆ 多系统之间的信息交换服务
- ◆ 多系统之间的互操作性服务
- ◆ 多系统之间的隔离、监管、认证、管理和控制服务
- ◆ 平台实现的宏观意义上的功能服务（例如，评分定价、流量控制等）

进行平台建设要注意以下几点：

- ◆ 平台是面向多系统和多代理的
- ◆ 面向可视化与代理化服务的平台
- ◆ 互操作性引发安全问题
- ◆ 远程服务的安全性问题

26.3 可信网络世界体系结构框架(TCAF)

基于上述分析，我们积极启动了可信网络世界体系结构框架（TCAF，Trusted Cyber Architecture Framework）标准化工作。可信网络世界体系结构框架企业联盟制定的标准是我国首次开发的信息化体系结构标准化产品系列。该体系结构标准化产品系列是针对信息化建设可信系统与平台的范例标准，是实践推荐标准，对我国当前的信息化，尤其是电子政务、电子商务、公众服务和

国家基础设施信息化建设非常重要。这个体系结构标准产品系列包括：

- ◆ 可信计算平台（TCP） 面向计算机系列（基础性）。
- ◆ 可信网络平台（TNP） 面向信息基础设施（基础性）。
- ◆ 可信管理平台（TMP） 面向数据与系统管理（基础性）。
- ◆ 可信认证平台（TCAP） 面向鉴别与认证（基础性）。
- ◆ 可信监管平台（TSP） 面向行为与内容监管（基础性）。
- ◆ 可信信息交换平台（TxP） 面向电子政务的应用平台建设。
- ◆ 可信应用平台（TAP） 企业级别的综合应用平台建设。
- ◆ 可信交易平台（TeCP） 面向电子商务的应用平台建设。
- ◆ 可信公众服务平台（TPSP） 面向公众服务的应用平台建设。
- ◆ 可信数据库访问管理平台 面向共享数据工程（TSHADEP）等可信平台的建设。

为什么说这些平台是可信的？主要是因为它们采取了如下的措施：

- ◆ 可信的信息基础设施系统。
- ◆ 可信应用系统。
- ◆ 可信接入 实现设备、用户和业务注册、鉴别、加密和撤销。
- ◆ 可信网关 使用活性标签和 Agent MAC 模式，实现系统、主体、客体之间的可信隔离。
- ◆ 可信认证 使用 CPK 活性认证，实现密钥管理。
- ◆ 可信管理 对数据、系统和网络的配置、定位、隔离、安全等实施管理。
- ◆ 可信监管 对系统、应用和用户实行行为与行为结果监管。

在 TCAF 计划中，要求实现表 26-1 到表 26-4 中描述的服务与

能力。从中可以看出，TCAF 计划的目标已经不仅是维护网络世界的防护能力，而主要是为人类信息化行为和网络世界的安全可信秩序的建立提供服务能力。

◆ TCAF 网络世界运营安全功能描述如表 26-1 和表 26-2 所示。

表 26-1 信息内容运营安全功能描述表

信息内容	提供服务与能力	说明
	大范围与大规模数据挖掘、检索	
	超海量票据监管	
	内容真实可信	
	内容认证	
	内容取证	
	内容监管	
	内容标签	
	内容危害	
	内容过滤	
	内容垃圾	

表 26-2 主体行为运营安全功能描述表

主体行为	提供服务与能力	说明
	身份认证	
	访问控制	
	主体权限、授权	
	主体安全标签	
	审计	
	入侵检测与监控	
	主体行为可信性	
	行为有效性	

(续表)

主体行为	提供服务与能力	说明
	行为完整性	
	行为保密性	
	行为连续性	
	主体定位与认证	
	网络主体追踪	
	行为取证	
	数字警察	
	代理执法	
	数字规划	
	虚拟企业	
	行为发现、检测	
	行为认证	
	行为监管	
	行为管理	
	行为控制	
	行为对抗	
	行为对抗态势测试与评估	
	行为保密	
	行为预警	
	可信管理平台	
	可信监管平台	
	可信认证平台	
	可信网络平台	
	可信交易平台	
	可信服务平台	
	可信应用平台	

(续表)

主体行为	提供服务与能力	说明
	可信交换平台	
	可信数据共享管理平台	

◆ TCAF 网络世界数据与系统的安全功能描述如表 26-3 和表 26-4 所示。

表 26-3 客体数据安全功能描述表

客体数据	提供服务与能力	说明
	数据保密性	
	数据完整性	
	信息隐藏	
	数字签名、印鉴	
	客体安全标签	
	信息标签	
	模式识别	
	数据备份与恢复	
	数据管理	
	标签路由与交换	
	信元定位：设备路径印鉴标签	
	安全分类标签	
	客体定位与认证	
	活性客体	
	活性标签	
	活性密钥	
	活性认证	
	信息交换标签	
	个体与群体完整性	

表 26-4 系统网络安全功能描述表

系统网络	提供服务与能力	说明
	系统完整性	
	系统可用性	
	反病毒技术	
	漏洞扫描	
	安全补丁、安全加固	
	计算机安全	
	操作系统安全	
	应用安全	
	系统配置安全	
	网络设备安全	
	网络协议安全	
	网络服务安全	
	网络系统的结构性安全	
	VPN	
	防火墙	
	系统与网络隔离	
	互操作性安全	
	远程服务安全	
	系统与网络防护系统	
	系统与网络监控系统	
	系统与网络管理系统	
	安全资源管理系统	
	系统备份与恢复	
	系统可信测试	
	系统完整性测试	
	设备定位与认证	

(续表)

系统网络	提供服务与能力	说明
	网络隔离监管	
	可信网关	
	可信接入	
	可信计算机	
	可信移动终端	
	系统伴侣、信道伴侣	
	可信安全管理平台	
	安全资源组网	
	管理、监管、控制单独组网	
	Agent MIS	
	网络系统的结构性安全	
	互操作性安全	
	远程服务安全	
	对象服务器	
	主体服务器	
	多代理可移动可生长群体系统	
	多代理群体计算系统	
	网格操作系统	
	安全体系结构	
	安全软件工程	
	Agent技术产业基地、AOP、AOSE	

第 27 章

信息化技术法规

国家、行业与地区制定信息化发展战略时，必须考虑制定相应的信息化技术法规体系。什么是信息化技术法规体系？技术法规或条例是推动用户技术标准实施以及维护信息化系统运营、管理和服务的正常秩序以及规定突发情况紧急处理方法的法规体系。

任何国家都有强制性的技术标准，这些标准对于规范信息化建设是非常必要的，例如，信息安全法规、互联网安全管理条例、测评认证法规、无线通信使用频率法规、电子签名法、电子印鉴法、电子标签法规、电子票据法、网络监控法和网络代理法规等。

技术法规对于用户标准化建设非常必要。例如，美国在 20 世纪 90 年代制定公共操作环境（DII COE）标准系列时，就有相关的技术法规与政府条例支持标准的建设：对所有为美国军方和政府服务的 IT 厂商发出告示，要求它们必须遵守 DII COE 标准的规定，同时要求它们同军方、政府签定长期服务合同（例如 15 年），如果它们不遵守这些规定，政府将不采购其产品。于是，为军方服务的各厂商都提供一种面向 COE 的产品系列，专门为军方采购服务，确保军方信息化系统互联、互通和互操作。同时，对于军方自己，也提出了前瞻性升级和换代的计划，要求在一定的时间内实现 COE 计划规定的内容。也就是说，将标准化的推行与政府采购法结合起来，强制推行标准化的实施。又例如，在通信领域规定标签路由和交换的标签技术法规，在推行中也可以采用与采购法结合的方法。

在美国和欧洲，信息化安全建设对于军方和政府是强制性的。例如，实行强制性的信息安全测评认证，实行电子监听法、监控法和网络代理法等技术法规。如果不制定这些法规，将会使在网络上维护安全的执法机构失去执法的依据。同时，这些法规还规范了执法的行为，避免或减少了执法过程中错误的发生，维护了国民的权益。技术法规维护信息化系统运营、管理和服务的正常

秩序，是一切正常进行的重要保障。

信息化建设中同样会遇到突发灾难事件，造成业务运营机构、管理机构和各种业务与技术组织瘫痪、停顿，造成信息系统不能正常工作，造成信息与数据的丢失。为了解决这些问题，应当重视业务连续性和安全应急体系的建设。不仅要在各企业范围内进行这种安全应急体系建设，而且要在行业与地方（城市）建立相应的体系或支援体系，开展信息系统的灾备业务，建立灾备中心和基地。同时，保险业应当与这些应急体系共同开展信息数据的保险业务，实现风险的规避、转移和化解。一方面，应当使安全应急体系的建设有标准可循，使其能够进行评估和监控；另一方面，也应当建立相应的技术法规，来确保信息化安全应急体系建设的投入、运营、管理和评估等工作的开展。

必须重视信息化技术法规的理论研究。信息化技术法规是信息化总体技术领域研究的重要内容，信息化技术法规的制定是建立在人类信息化行为学研究基础之上的。由于人类信息化行为的基础理论研究欠缺，所以，在制定信息化技术法规时缺少理论支持——信息化技术法规是依据经验和各种案例制定的。这种在缺少理论或者形式化方法支持的情况下制定的信息化技术法规很难在网络中执行。

犯罪行为在网络内部，执法却在网络外部，因而造成了一系列的矛盾和困惑。因此，应该制定出一种管理人类信息化行为和网络世界行为的技术法规，使这种法规的执行在网络中。

第 28 章

信息化标准化理论

28.1 标准需要理论与体系结构的研究

标准化工作对于许多国家来说是建立技术壁垒的重要手段。在进行标准化工作时，要考虑国际、国家、地方、领域和企业多方面的发展与利益平衡的问题。建立标准最终要为建立和谐的世界服务，为世界和国家各方利益的平衡服务，不是一个单纯的技术问题。在进行标准化的总体规划与设计时，必须从不同的层次和方面进行综合考虑。例如，我们可以在制定信息化标准时考虑如下的原则：

1. 运营标准的定位原则

- ◆ **国际运营标准原则** 所谓国际运营标准原则，是指运营体系在国际范围内进行，虽然运营体系中存在着各国内部的运营体系，但是这种运营概念必须关注各国运营体系的互联、互通和互操作。在这个国际范围的运营体系中存在着广泛的合作，各国在全球化的发展过程中，都会利用技术标准建立技术壁垒，实现国家利益的保护。其中有对抗、竞争、博弈、合作等复杂情况。如果过分强调自己的利益，必然会为各方和自己带来麻烦，反而适得其反。所以，当今世界的国际范围内的运营技术标准是面向多运营模式以及多运营与系统体系结构的，是跨国合作的多技术融合标准，支持世界范围内的多厂商产品，承认发展历史，共同面向未来。例如，在网络基础设施中，为了实现全球网络的运行，支持这种运行的技术标准应当是国际范畴内的标准。通常，这类标准属于公共服务类的运营。
- ◆ **国家运营标准原则** 所谓国家运营标准原则，是指运营体

系必须实现国家主体概念下的控制运营，其运营是国家主权。显然，这样的运营体系的技术标准必须由国家制定，是国家主权或者国家范围内的事务需要制定的标准，例如，涉及到主权、社会、文化、环境、安全的技术标准。在信息化中，专门服务、安全标准、监管标准、管理标准、密钥标准、评估标准、测试标准等统称为政府内部服务和政府监管职能的技术标准。例如电信与金融行业，在 WTO 的规则下，国外企业期盼进入到这些服务业市场。电信与金融的服务运营将会参与到国际范围的运营标准中。但是，电信与金融运营的安全、监管、认证等方面的工作属于政府的监管职能，其标准是国家性的，而且针对国内所有运营企业（包括国外企业）。

- ◆ **企业运营标准原则** 这种标准仅在企业内部使用，标准的制定属于运营企业的工作权利和范围。例如，企业的业务管理和各种管理的标准，基本上属于企业的运营标准。企业运营的技术标准同样是面向企业运营体系结构和系统体系结构的。

2. 技术标准的体系结构设计原则

- ◆ 技术标准的体系结构是面向运营体系结构和系统体系结构的，称之为技术体系结构，确立技术标准的引用模型（TRM）。
- ◆ 技术标准面向体系结构的设计就是要确立技术体系结构的组成，确立技术标准之间的合作特性和发挥综合效益的原则。

3. 技术标准的互操作性产业联盟原则

技术融合与产品的互操作性标准是产业共同利益要求制定的标准，是市场中技术存在形式的要求，实现技术与产品的互联、互通和互操作。

4. 技术服务功能的企业竞争性标准原则

- ◆ **产业技术标准原则** 标准化组织制定的信息技术标准，代表着一种技术体制和服务模式。技术标准是竞争力标准。
- ◆ **企业技术标准原则** 产品的技术标准，代表着产品的质量、技术水平、服务承诺。产品标准是竞争力标准。

正因为如此，制定标准需要进行总体或体系结构的研究，要对整个标准化工作有整体的考虑与研究。例如，要考虑安全标准，对已经具有的技术体制和产品标准，仅仅提出要求改变其不安全性或者脆弱性，这种标准的首先执行者是原有技术与产品的供应企业，新的企业进入其中，必然会遇到合作的问题，否则就要研究新技术与产品。尤其对于其他国家的技术与产品，这种安全标准将会遇到非常大的麻烦，例如 CPU、操作系统和 BIOS 等基础产品。如果不认真研究和考虑，那么这种安全标准是为别人制定的。从作者的工作体会看，把安全标准制定为非产品捆绑的标准是必须考虑的原则，例如，提出操作系统、CPU、BIOS 等产品的可信测试标准（比较 TCP 标准中的 TPM 系统完整性测试）、技术产品非预期行为的可信监管标准、产品资源配置与管理标准、网络服务安全技术标准等。这样做的好处是首先把这些基础技术产品列入非可信的对象范畴，纳入可信测试与监管的范围，而可信测试与监管是国家制定标准的明确范畴，把基础技术产品作为开放形式的产品，纳入国际标准的范畴内。如此制定标准，成本

低，操作性好，麻烦也少。反过来，对基础产品捆绑安全服务相关内容，国家还可以明确采取干预措施。信息化要讲科学，制定标准也要讲科学。标准化讲科学，就是讲效益。

标准化工作要进行体系结构研究的意思是：

1. 标准，尤其是用户标准体系，是面向体系结构的。信息化的体系结构包括运营体系结构、系统体系结构和技术体系结构。其技术体系结构是面向系统体系结构和运营体系结构的，而不是脱离运营体系结构与系统体系结构的。技术体系结构就是确立其技术标准体系。

2. 设计技术体系结构或者标准体系，首先要根据需要与可能，确立系统功能描述点的集合与运营功能描述点的集合，并在上述两个集合范围内进行所有功能的定义。功能描述点集合包括业务功能、互操作性功能、安全功能和服务功能等。注意，技术体系结构的定义对于各种应用领域是不同的。

3. 设计标准体系可以考虑分级标准，但是这种考虑是面向应用领域的，例如政府、金融、电信、电力、财政、交通、民航等领域。这种分级标准实际上是面向应用领域的系统体系结构的系统与分系统，分级定义包括功能配置分级定义、功能强度分级定义、主体行为分级定义和互操作性分级定义，具体描述如下：

◆ **主体行为分级标准定义** 针对运营体系结构的所有主体进行行为和行为结果的标准定义。确立行为模式、行为流程、行为过程和行为协议，并根据这些行为模式确立行为控制、行为监管、行为认证、行为管理，甚至行为对抗，进行主体、行为、客体的预期行为规则标准的定义。同样应当实行分级标准定义。

◆ **功能配置分级标准定义** 依照系统功能描述点的集合与

运营功能描述点的集合进行功能描述点配置分级和系统功能强度强度分级设计。功能描述点的配置分级描述包括拓扑结构分配和层次结构配置两个方面，根据分级的要求进行分级设计。

◆ **功能强度分级标准定义** 依照系统功能描述点的集合与运营功能描述点的集合进行系统功能强度分级设计。所谓功能强度分级，是针对同一类型的功能，考虑实现技术的差异，评价其不同的强度级别，根据分级的要求进行分级设计。

◆ **互操作性分级标准定义** 针对系统体系结构的所有系统和分系统进行互操作性标准定义，包括定义与引用关系、接口关系（API）和外部环境的接口（EEI）的标准定义或规定。互操作性同样应当进行分级定义。

信息化标准化理论属于网络世界行为学的研究内容。技术标准的贯彻与执行是通过测评认证来实现的，而不是通过下发一个标准化文件来实现的。通过下面的讨论，我们便可以理解，为什么必须强制性地执行这些技术标准。为了能够在整个企业范围内贯彻实施，就必须制定相应的技术法规，来确保技术标准的执行。例如，在银行信息化建设中，必须实行支持互操作性的安全性标准与规范，不执行互操作性就不可能实现整个银行信息系统的一体化运行，就不可能实现银行范围的综合管理，也就不可能实现银行领导的信息化管理。银行信息化实现了互联、互通和互操作之后，信息系统安全性便会成为十分重要的问题，否则，银行一体化的信息化就会失去安全保障。

28.2 用户标准体系

1. 什么是用户标准体系

用户标准是行业用户为实现信息化而制定的技术标准系列，主要包括信息化体系结构标准、互操作性标准、网络服务标准、安全标准、监管标准、评估标准和测评标准等。这些标准不是企业的产品与技术标准。在很长一段时间内，信息技术标准化工作主要是面向开发和生产的，例如，软件工程整个生存期要求的技术标准。企业对信息技术的标准更感兴趣，而用户知道选用符合标准的产品便可以了。但是，用户有时长期得不到符合国际标准的产品，而得到的往往是符合企业标准的产品。信息技术标准（包括国际标准化组织制定的标准）基本上都是从企业角度考虑的。

用户标准是由行业用户出面牵头，在标准化专业机构的配合下，在行业产品、系统和服务供应商的参与下，在保护用户信息技术应用系统建设投资效益的基础上，要求企业实施长期成熟发展、参与竞争、提高服务质量、降低服务成本、实现信息产品综合功能效益的标准体系，这就是用户标准体系。用户关心的标准实际上是用户问题的“解决方案”标准。

2. 用户标准体系分类

用户标准体系分为信息化体系结构标准、互操作性标准、网络服务标准、安全标准、监管标准、评估标准、测评标准和服务等级标准等。其中，体系结构标准是最重要的，包括运行体系结构标准、系统体系结构标准和技术体系结构标准。

我们虽然做了一些工作，但是对于信息化需要建立用户技术标准的认识还是非常初步的，还需要进一步提高和加强。

28.3 现代标准化是一种网络服务体系

谈到用户标准体系的建设，传统意义上认为，标准化是对应用模式、系统结构和技术体制的一些质量方面的规定，是以文档的形式发布的，期望用户和企业共同遵守。但是，这些标准被遵守了吗？为了检查或评估标准的执行情况，需要针对这些标准或准则建立测评认证机构，通过认证证书管理与控制产品质量标准的执行。传统的测评认证过程是：企业在开发完成产品后，通过正式申报的方式向测评认证机构申请，被受理后，企业将所开发的产品及其文档交付测评认证机构进行测评认证，包括产品开发环境、研读文档、确认产品质量、测试功能、测试性能、文档与实现的一致性检查、确认使用环境等多项技术与质量标准。这些活动都完成后，还要进行不同级别的评审，最后才能决定是否通过测评认证。如果通过测评认证，则颁发证书，并向社会发布认证情况；如果没有通过测评认证，则提出整改意见，企业拿回产品进行修改，并进行二次申报。应该说，企业为通过测评认证还是要付出较大代价的，需要的测评时间往往很长。所以，信息化的测评工作需要进入到网络中，通过测评业务的网络化服务，在网络中进行真实的测评工作。

第 29 章



信息化评估理论

围绕着测评的有各种评估标准，例如质量评估标准、各种软件工程评估标准和成熟能力评估标准等。同时，还存在着针对信息技术企业的评估。

29.1 信息化工作的价值评估是核心

目前，人们都喜欢对评估对象的过程、目标、任务和管理等方面开展评估工作。但是，这些内容的评估实际上干预了企业的自主工作，影响了企业评估工作的创新、竞争力与积极性。正确的做法是把评估的重点放在评估信息化工作的价值上，即评估对象的信息资产，如同我们在信息化监管理论中讨论的那样。评估工作要出台一些标准和准则，最重要的是给出信息资产价值化评分与定价的标准以及信息资产风险价值化评分与定价的标准。除了实行价值评估外，还应当对评估过程给出原则性要求的规定，例如，给出必须做的事情的规定、评估质量的规定和评估文档的规定，而不要给出方法和功能性的硬性规定，因为方法、流程和功能性评估是企业的权利，应该留给企业。图 29-1 显示了信息化价值评估的结构。

29.2 评估有效性的再评估

评估工作要讲效益，要实行效益或有效性的再评估。没有有效性的再评估，就可以认为评估工作不讲效益，或者说这种评估不负责任。有效性评估对以前进行的价值评估进行有效性的评估，如图 29-2 所示。

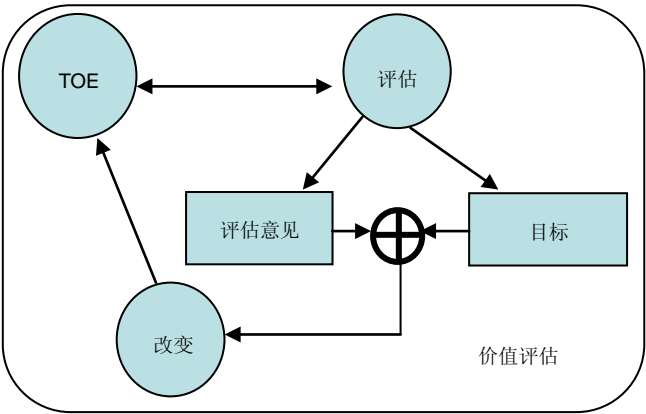


图 29-1 信息化价值评估结构

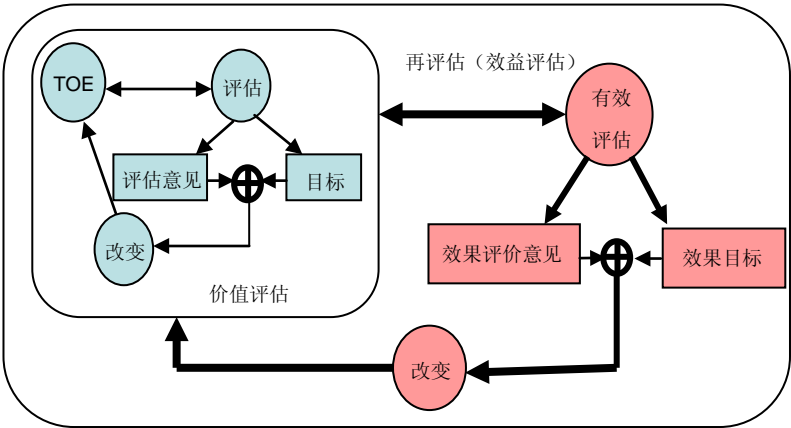


图 29-2 信息化有效性评估结构

虽然对信息资产与风险的价值化进行评估是最核心的工作内容，但是，不同的关注者对企业风险的关注点和采用的评估方法是不同的。例如，银行家讨论的风险包括金融政策风险、货币风险、信用风险、操作风险和市场风险等，采用巴塞尔资本协议作为评分和定价的评估方法；信息领域的科学家主要关心的是信息安全、信息系统安全、网络安全、通信安全、系统与网络服务行为的安全和系统的运营及管理方面的风险问题，同时还关心系统的可靠性和连续性服务等问题，采用系统工程的风险评估方法。正因为如此，面对的问题不同，评估方法也不同。

风险总体评估实际上就是对信息资产进行评分和定价。为此，需要给出所有风险评分和定价指标体系，并将这种评分和定价指标体系在行为监管的每一个环节中实现。业务风险监管在研究了对什么行为实行监管之后，还必须针对已经实现监管的行为，进一步分析监管的特性，例如，考查行为可信性（行为预期）、行为有效性（行为价值）、内容真实性、行为完整性、行为保密性和行为连续性。要求给出行为特性监管的度量方法。为了便于给出评分与定价的方法，要区别业务行为与技术行为。下面，我们以可信性评估与定价为例来讨论行为可信性评估问题。可信性是指主体行为历史记录反映主体行为有没有违约、违规、违法、无依据、超越权限、超范围以及破坏数据、信息、行为完整性和保密性等统计特性，例如违约概率和错误概率等。

例如，业务行为可信性的评分方法如下：

$$\text{可信性评分} = \text{非预期行为的时间与总办公时间的比值乘以行为可信性的权重值} + \text{行为评估评分值} + \text{行政处罚评分}$$

管理定价指标体系是根据企业各种业务职务和角色，在整个企业范围内制定相互配合的工作定价。制定企业管理平均定价体系是实现企业管理会计、绩效考评的基础指标体系，也是企业行为监管的落脚点之一。至少应在如下几个方面建立相应的定价体系：企业员工每小时工作时间平均定价、业务信息系统每小时运行时间平均定价、终端运行单位时间平均定价、网络运行单位时间平均定价、有效工作时间平均百分比、风险定价和损失定价。需要说明的是，上述定价体系都是面向角色的。

这种评分与定价完全可以在计算机与网络系统中实现，即信息资产风险评分与定价可以在网络世界中实现。这种在网络世界中实行的评估方法与在网络世界中实行的监管方法是一致的。

第 30 章

信息化测评认证理论

信息化测评认证技术是信息化总体技术领域的重要组成部分。目前,我国开展的信息化测评认证比较多,但比较乱;有的测评认证是面向产品的,有的测评认证是面向系统的,有的测评认证是面向服务的;有计算机的系统事务处理能力的基准测评、网络设备的性能与功能的测评、软件功能的测评、信息安全技术测评、互操作性测评和可靠性测评等。

标准化控制实际上是一种质量控制,在实际工作中通常采用两种方法:

- ◆ 其一,采用上述技术体制满足的强制要求。用户,尤其是大用户,可以采用内部技术法规的形式,以文件的形式声明不满足要求、不能中标和不能采购的规定,其中要特别强调管理否决权的实际意义。
- ◆ 其二,有计划地实施测评认证,保证技术体制的落实。这些测评包括互操作性测评、安全保障测评、安全监管测评、安全应急预案测评、代理服务测评和功能性能强度测评等。

信息安全测评认证的标准很多,下面列出了主要的几个(它们有不同的侧重点,也有不断进步完善的因素):

- ◆ ISO/IEC 15408 (CC) 信息技术产品的安全测评认证
- ◆ ISO 13335 安全管理标准
- ◆ 英国与北欧制定的 BS7799 测评模型和框架以及 PD3000 标准 (ISO 17799)

信息技术安全测评包括对安全目标、安全风险、安全策略、安全功能、开发环境和应用环境的测评认证。

30.1 信息技术的安全测评认证

信息技术安全功能测评可以分为 5 类，下面分别进行介绍。

1. 安全功能的标准测评

目前，国际上具有多年历史的权威信息安全评估标准有美国国防部公布的可信计算机系统评估准则 TCSEC（彩虹系列标准）、欧美等国家提出的 ITSEC 准则和 CC 标准以及国际标准化组织在 CC 基础上提出的 ISO/IEC 15408 信息技术安全评估准则。我国的安全标准参照 TCSEC、ITSEC 和 CC（ISO/IEC 15408 评估准则）标准实行。将来，评估准则会统一到 ISO/IEC 15408 评估准则上。1985 年美国国防部提出的 TCSEC 是世界上第一个信息系统安全的评估准则，其意义巨大，TCSEC 和 TCSEC 的网络解释（TNI）与数据库系统解释（TDI）依然对安全防护具有指导意义。

对于 TCSEC 的测评认证标准，其安全测评的原理可以用如图 10-2 所示的三维图表示，其三维分别是安全功能、系统结构和安全级别。

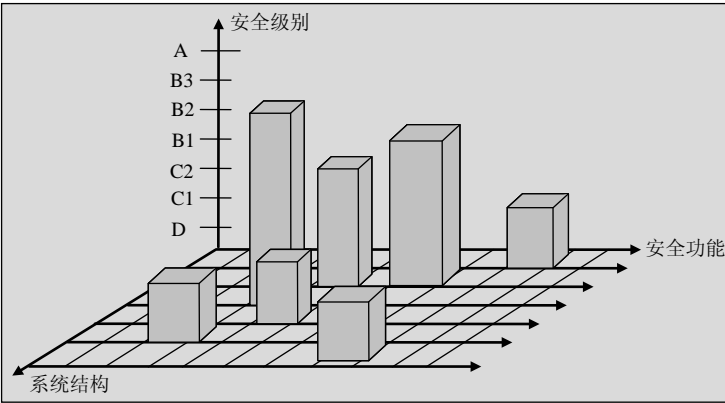


图 30-1 TCSEC 的系统结构、安全功能和安全级别

对于操作系统来说,系统结构可以是如下的内容(主体、客体):操作系统的安装、生成、配置和优化服务,操作系统的连接服务,操作系统的中断和控制服务,操作系统的进程调度与管理服务,操作系统的进程同步与进程间通信服务(IPC),操作系统的存储管理系统服务,操作系统的 I/O 系统服务,操作系统的文件系统服务,操作系统的多 CPU 并行处理服务,操作系统的网络系统服务,操作系统的通信系统服务,操作系统的屏幕管理服务,对象与可视化系统服务,操作系统的系统管理服务,操作系统的系统安全服务,操作系统的异常处理服务,操作系统的开发支撑服务和操作系统的应用支撑服务等。

安全功能包括:自主访问控制(DAC)(访问控制策略)、强制访问控制(MAC)(信息流控制策略)、安全标识、客体重用、可信通道、可信通路、标识与鉴别(I&A)、安全审计、安全测试、隐蔽通道、系统结构、系统完整性、可信恢复、可信分布、可信设备管理、配置管理、设计说明验证和可信文档。

安全级别是 D, C1, C2, B1, B2, B3 和 A。但是,其应用有局限性。这些局限性表现在:

测评由厂商提出申请,测评认证机构 NCSC 受理后,签署测评认证合同,厂商根据申报产品安全测评的安全级别,依据标准级别规定的提交文档要求,将文档和产品提供给测评认证机构。接着,测评认证机构提出厂商必须回答的问题,包括体系结构、系统组成、指令系统、CPU 结构、存储器、外部设备、操作系统组成、TCB 结构、自主访问控制(DAC)(访问控制策略)、强制访问控制(MAC)(信息流控制策略)、安全标签、客体重用、可信通道、可信通路、标识与鉴别(I&A)、安全审计、安全测试、隐蔽通道、系统结构、系统完整性、可信恢复、可信分布、可信设备管理、配置管理、设计说明验证和可信文档等,厂商必须一一做

出书面正式回答，作为产品安全评估的依据之一。测评认证机构为此产品组织测试组，开始实施针对性的测试工作。首先，确认厂商宣称的产品安全功能是否属实，了解其采用的技术和方案，这个测试称为安全功能测试。其次，实施攻击性测试，其测试方法称为“漏洞假说方法”，实施手动和程序攻击。最后，测试组写出规范的测试报告，并将其交给认证部门供进一步的评估使用。

虽然 TCSEC 安全测评准则为信息安全建设发挥了开创性的作用，但是它现在已经过时了，已经不能满足信息技术安全建设和管理发展的需求，对于测评开放系统的安全已经远远不够了。其理由如下：

- ◆ TCSEC 是针对建立无漏洞和非侵入系统制定的分级标准。TCSEC 的安全模型不是基于时间的，而是基于功能、角色、规则和代数格等空间与功能意义的。TCSEC 是针对单一计算机，特别是小型计算机和主机结构的大型计算机制定的测评标准。
- ◆ TCSEC 主要用于军事和政府信息系统，对于个人和商用，采用这个准则是有困难的。也就是说，其安全性主要是针对保密性而制定的，对完整性和可用性的研究不够，忽略了不同行业计算机应用安全性的差别。例如，有些系统对完整性的要求很高，而对保密性的要求较低；有些系统对服务连续性和拒绝服务攻击的要求较高；有些系统对系统资源可用性要求较高；等等。TCSEC 的安全概念是静态的，还表现在安全策略是固定的，安全策略不能针对不同的安全威胁实施相应的组合。应当将安全策略的制定与安全目标和安全环境相结合。
- ◆ TCSEC 的三个 B 标准级别，采用强制访问控制（MAC）

和安全标识或安全标签技术,受到了业界的关注。具有 B 安全级别的操作系统,由于要求对已经开发与运营的应用系统进行修改,而且 B 安全级别的操作系统开发应用系统的成本要远远高于 C2 安全级别的操作系统开发应用系统的成本;从全世界看,这种安全操作系统概念得不到市场的认可与承认。但是,这种安全标签的强制思想,在网络中获得了成功应用,例如 HP 的 VVOS 在网关中的应用。目前,TCSEC 的网络解释缺少成功实践的支持,尤其是对于互联网和商用网络,缺少成功实践的支持。

- ◆ 美国 NSA 测评一个安全操作系统需要花一两年甚至更长的时间,这个时间已经超过了目前一代信息技术发展的时间。也就是说,TCSEC 测评的可操作性较差,缺少测评方法、框架和具体标准的支持。

基于对 TCSEC 的批判,考虑计算机商用的安全问题,例如金融应用,提出了许多军用可信信息系统所没有的安全问题,尤其是网络安全问题。尽管 TCSEC 有一个网络解释,但它是针对军用可信网络的,不适合商用的非可信网络。进入 20 世纪 90 年代后,信息安全在美国得到了普遍的重视。人们认识到需要研究新的测评认证准则,于是,新的准则 ITSEC 出台了。ITSEC 是 Information Technique Security Evaluation Criteria (信息技术安全评估准则)的简称,是英国等欧洲国家继美国推出 TCSEC 之后联合制定的信息产品安全评估标准。该准则于 1989 年提出,到 1991 年批准,直到 1996 年才完善了全部细则。ITSEC 把准则划分为结构准则和操作准则。结构准则又划分为 4 个方面:功能的可用性、功能的捆绑、机制强度和结构脆弱性评估;操作准则划分为两个方面:易用性和脆弱性。ITSEC 把安全级划分为 E1, E2, E3, E4, E5 和

E6 共 6 个级别。

每一个安全级别把结构评估准则划分为：

- ◆ **开发过程** 需求、体系结构设计、详细设计和实施。
- ◆ **开发环境** 配置控制、程序设计语言和编译、开发者安全。
- ◆ **操作文档** 用户文档、管理文档。
- ◆ **操作环境** 递交与配置、安装与操作。

显然，ITSEC 评估准则不只评估产品本身，而且还评估开发过程和操作，强调安全的全过程性。由于 ITSEC 相对于 TCSEC 评估准则来说，评估程序比较简明，评估速度快，适应厂商的需要，所以参加 ITSEC 评估的产品和厂商比参加 TCSEC 评估的要多。

ITSEC 的评估级别和 TCSEC 的评估级别之间没有简单的对应关系，但是因为业界已经熟悉 TCSEC 了，所以 ITSEC 还是制定了与 TCSEC 等价的功能关系，例如，F-C1，F-C2，F-B1，F-B2，F-B3，F-A 就是与 TCSEC 的 C1，C2，B1，B2，B3，A 相对应的。另外，ITSEC 还定义了 F-IN（强调完整性）、F-AV（强调可用性）、F-DI（强调数据交换完整性）、F-DC（强调数据交换保密性）、F-DX（强调数据交换保密性和完整性）等，与 TCSEC 没有直接关系。虽然与 TCSEC 相比，ITSEC 有一定的优势，但是，它的进步依然是有局限性的：它实际上是在 TCSEC 的基础上加上数据交换和资源可用性的安全特性，考虑开发环境 and 应用环境，它依然是静态的，而不是基于时间的。因此，从体制上讲，ITSEC 与 TCSEC 是相同的。

信息安全的本质是管理，而管理的关键是管理自动化，信息安全应当是综合的，既是面向空间、时间和功能的，也应当是面向人员的、全方位的信息安全模型。随着对信息安全的本质的理

解，美国、欧洲和加拿大等 7 国联合制定了 CC (Common Criteria) 安全评估准则，到 1999 年，CC 测评认证标准被国际标准化组织接受为国际标准，命名为 ISO/IEC 15408。这个由国际标准化组织 (ISO) 和国际电联共同制定的、于 1999 年得到批准的信息技术安全评估准则由如下三部分所组成：

- ◆ 第一部分为介绍和一般模型。
- ◆ 第二部分为安全功能需求。
- ◆ 第三部分为安全保证（保障）需求。

ISO/IEC 15408 准则比以往的信息技术安全评估准则更加规范，采用类 (Class)、保证族 (Assurance Family)、保证部件 (Assurance Component) 和保证元素 (Assurance Element) 的方式定义基本准则，类别中有若干族，族中有若干部件，部件中有若干元件。在新的 ISO/IEC 15408 测评认证标准中，对安全功能的强度不再用 D, C1, C2, B1, B2, B3 和 A 来表示，而是用基本强度、适中强度和高强度来表示。新的测评认证标准不仅对产品的安全功能进行测评认证，同时还对开发过程、开发环境以及应用管理与使用环境进行测评认证。测评认证是面向需求的，针对威胁的，其流程如图 30-2 所示。

图 30-3 用两个三维图显示了 CC 安全评估准则的原理。对于计算机网络系统来说，图 30-3 中的系统结构可以是如下的内容：物理层、数据链路层、网络层、传输层、会话层、表示层和应用层；对于 TCP/IP 网络而言，则是物理层、数据链路层、IP 层、TCP (UDP) 层和应用层。

图 30-3 中的安全功能可以包括：安全审计、通信安全、加密支持、用户数据保护、标识与鉴别、安全管理、隐私保护、TSF 保护、资源利用、TOE 访问和可信通路等。注意，CC 的安全功能

分类与 TCSEC 不同，但 TCSEC 中的自主访问控制（DAC）（访问控制策略）、强制访问控制（MAC）（信息流控制策略）、安全标识、客体重用、可信通道、可信通路、标识与鉴别（I&A）、安全审计、安全测试、隐蔽通道、系统结构、系统完整性、可信恢复、可信分布、可信设备管理、配置管理、设计说明验证和可信文档等均包含在 CC 中。

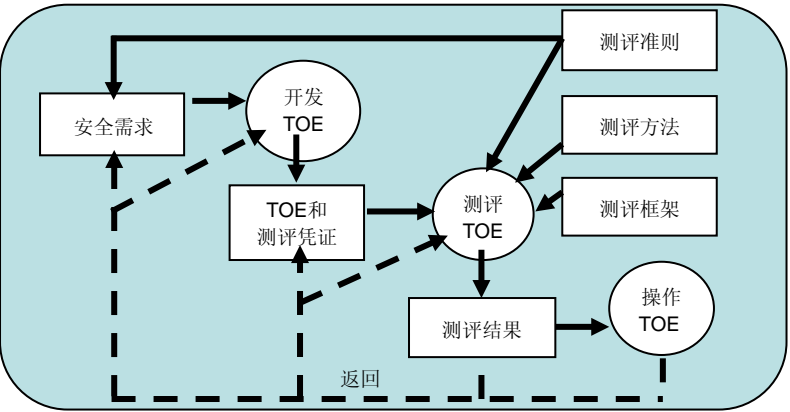
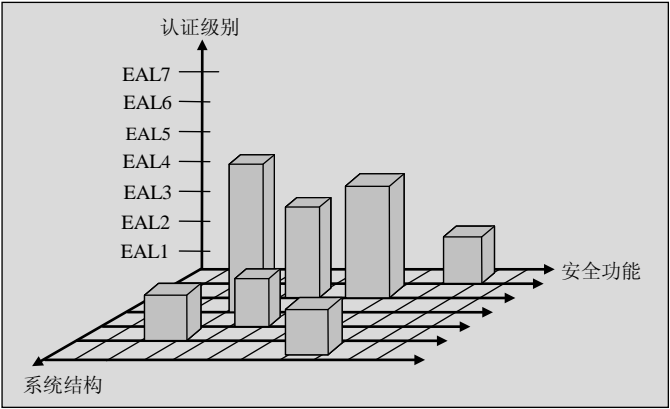
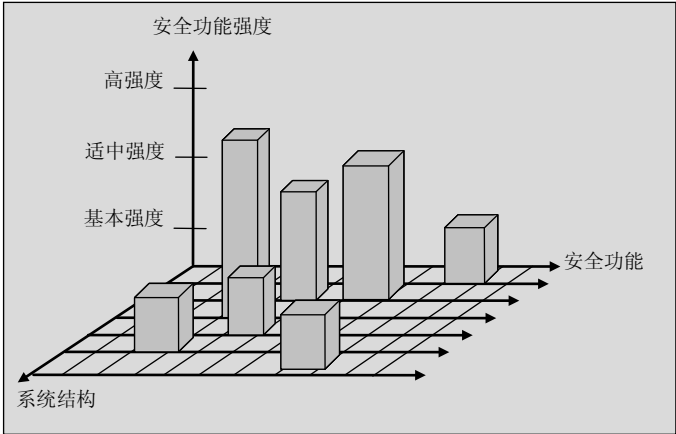


图 30-2 CC 测评流程图



(a) CC 的系统结构、安全功能和认证级别



(b) CC 的系统结构、安全功能和安全功能强度

图 30-3 CC 安全评估准则

从标准化的结构与概念定义中可以看出，CC 更加严格，概念更加清晰。TCSEC 标准中定义的安全评估级别将两个概念——质量标准和**安全功能标准（安全功能强度标准）混在一起，在 D，C1，C2，B1，B2，B3，A 不同级别的定义中，质量与安全功能同时随着级别的升级而升级，必然会造成高安全级才有高质量，而低安全级就没有高质量的情况。而在 CC 标准中，将这两个概念分离了，一个是质量标准的级别：EAL1~EAL7 共 7 个质量级别；另一个是安全功能的 3 个强度（基本强度 SML1、中等强度 SML2 和高强度 SML3），并对安全功能的强度提出了一个基本的计算和评估方法。安全功能的强度是可以“与时俱进”的。从标准化的体系结构概念上看，CC 更加科学，同时，不会再出现高质量才有高安全级的情况。

CC 标准更加重视全过程的安全问题，在设计、开发、生产、使用、维护等环节中强调安全问题。CC 标准是一个框架标准，是

面向用户信息安全需求的，是针对威胁的。对于具体产品和技术
的认证，CC 规定必须制定类别的安全防护结构测评（PP）标准和
具体的安全目标评测（ST）标准。在实施测评时，PP 标准 和 ST
标准非常重要。

安全防护结构测评（PP）从结构的角度出发，综合考虑其防
护结构（框架）的安全需求是完全的、一致的和健全的。在此评
估中，要评估其安全动机和安全需求理由，要给出安全结构能够
对抗信息安全的威胁级别（错误使用、蓄意直接攻击和精心设
计的有组织攻击）要求及安全机制强度要求。PP 测评带有类属性质，
一类安全产品（例如防火墙、路由器等）具有类属产品的标准性
质，提出 PP 测评的往往是领域和行业组织。例如，我们可以提出
网络远程服务序列的 PP 标准（网络远程数据访问 PP 标准、网络
远程安装服务安全 PP 标准、网络远程配置服务安全 PP 标准、网
络远程检测服务安全 PP 标准、网络远程监控服务安全 PP 标准、
网络远程维护服务安全 PP 标准、网络远程备份服务安全 PP 标准、
网络远程恢复服务安全 PP 标准、网络远程管理服务安全 PP 标准、
网络支付服务安全 PP 标准等），还可以提出 CPU 芯片安全 PP 标
准、IC 卡芯片安全 PP 标准、计算机/服务器安全 PP 标准、操作系
统安全 PP 标准、通信基础设施安全 PP 标准、网络设备安全 PP
标准、数据库管理系统安全 PP 标准、应用信息系统安全 PP 标准、
防火墙与门卫安全 PP 标准、网络协议与服务安全检测 PP 标准、
网络安全检测安全 PP 标准、网络安全监控安全 PP 标准、系统安
全扫描安全 PP 标准、系统安全监控安全 PP 标准、系统蓄意代码
安全 PP 标准和系统配置检测安全 PP 标准等。

安全目标测评（ST）从目标的角度出发，综合考虑其防护结
构（框架）的安全需求是完全的、一致的和健全的。在此评估中，
要评估其安全动机和安全需求理由。要给出安全结构能够对抗信

息安全的威胁级别（错误使用、蓄意直接攻击和精心设计的有组织攻击）要求及安全机制强度要求。ST 测评是实例性的，针对某一类属的特定产品，例如某厂商的防火墙产品、路由器产品等，提出 ST 测评的是产品的厂商。正由于如此，CC 标准通过这些 PP 标准和 ST 标准建立标准与被测评对象的针对性。测评与用户需求密切相关。

需要特别注意的是，ISO/IEC 15408 信息技术测评准则还有一个信息技术测评通用评估方法（CEM），它是针对 ISO/IEC 15408 IT 测评准则的。针对测评类别、部件和元件，在 CEM 中定义了行为（activity）、子行为（sub-activity）与活动（action）。对 PP 测评、ST 测评和 EAL 测评都提出了测评行为、子行为和活动的要求。

ISO/IEC 15408 IT 测评准则测试类别（CLASS ATE）包括覆盖、深度、功能和独立测试 4 个族，属于测评的测试部分。

ISO/IEC 15408 IT 测评准则的脆弱性评估类别（CLASS AVA）包括隐蔽通道分析（CCA）、错误使用（MSU）、安全功能强度（SOF）和脆弱性分析（ALA）4 个族，属于测评的评估部分。其中，脆弱性分析是比较复杂的，与攻击潜能测评联系。

在做脆弱性分析时，采用的分析部件不同，表示测评对象抵抗攻击的强度不同。例如，采用 VLA.3 部件，是研究抵抗中度攻击强度的威胁，对高强度攻击不能抵抗；采用 VLA.4 部件，是研究抵抗高度攻击强度的威胁，对无实践性的超高强度攻击不能抵抗；具体情况见表 30-1。

IAO/IEC 15408 第二部分中描述的安全功能针对其每一种功能，都要进行强度评估，而这种强度评估是与攻击的水平相关的（参见表 30-2）。例如，采用“SOF-高”功能强度分析，是研究抵抗高度攻击强度的威胁，对无实践性的超高强度攻击

不能抵抗。

表 30-1 脆弱性分级表

脆弱性分析部件	TOE 抵抗攻击的强度	TOE 不能抵抗攻击的潜能
VLA.4	高	无实践性
VLA.3	中	高
VLA.2	低	中

表 30-2 防护功能强度表

SOF 强度	防护抵抗攻击的强度	防护不能抵抗攻击的潜能
SOF-高	高	无实践性
SOF-中	中	高
SOF-低	低	中

强度评估与攻击的水平相关，攻击的水平又与什么因素相关呢？在 CEM 中对识别与实施攻击规定了 5 种因素：消耗时间、专门技术、TOE 设计和操作知识、TOE 访问和采用的设备，如表 30-3 所示。

表 30-3 功击水平因素表

	识别因素	实施因素
1	识别时间	实施时间
2	专业技术	专业技术
3	TOE 设计和操作知识	TOE 设计和操作的知识
4	TOE 访问	TOE 访问

(续表)

	识别因素	实施因素
5	分析的硬件/软件或其他专用设备	实施的硬件/软件或其他专用设备

下面，对这些因素分别进行解释：

- ◆ **消耗时间** 攻击者识别攻击对象和实施一次攻击的连续时间，攻击者攻击消耗的时间越长，说明抵抗攻击的能力或安全功能强度越高。
- ◆ **专业技术** 攻击者对 IT 专业知识了解程度是分析脆弱性的重要因素。显然，要求专家才能攻击的产品比外行人员就可以攻击的产品抵抗攻击的能力或安全功能强度要高。
 - **专家人员** 熟悉算法、协议和硬件结构，以及产品采用的安全概念、规则和方法。
 - **职业人员** 熟悉产品和系统的安全性行为和特点。
 - **外行人员** 不了解被攻击对象的专业知识和安全性能。
- ◆ **TOE 设计和操作知识** 区别于专业知识，是对 TOE 具体对象的设计和操作知识的了解水平。要求对 TOE 有深入了解才能攻击的产品比不了解 TOE 就可以攻击的产品抵抗攻击的能力或安全功能强度要高。
 - 对 TOE 不了解。
 - 了解 TOE 的一般性知识。
 - 了解 TOE 的敏感性知识。
- ◆ **TOE 访问** 攻击需要对 TOE 的脆弱性十分了解（包括技

术和管理方面)。一般来说,需要一定量的 TOE 访问(攻击者可以脱机准备攻击程序),访问的量也用时间来度量。实施攻击需要访问 TOE 的时间越长,说明抵抗攻击的能力或安全功能强度越高。

◆ 采用的设备 攻击者在攻击时采用的设备的水平是威胁的重要因素。需要使用专用设备实施攻击的产品比使用标准设备就可以攻击的产品抵抗攻击的能力或安全功能强度要高。

但是,从攻击的威胁来看,抵抗个体单点的攻击和抵抗群体的分布式协同攻击,是绝对不同的。

如何定量地对这些因素进行分析,在 CEM 中是有规定的,其具体内容如表 30-4 所示。

表 30-4 攻击能力评分表

因素	范围	识别值	实施值
消耗时间	<= 0.5小时	0	0
	< 1 天	2	3
	< 1 月	3	5
	> 1月	5	8
	无实践	*	*
专业技术	外行	0	0
	职业	2	2
	专家	5	4
TOE知识	无知	0	0
	公开	2	2
	敏感	5	4

(续表)

因素	范围	识别值	实施值
访问TOE	<= 0.5小时或不可检测	0	0
	< 1天	2	4
	< 1月	3	6
	> 1月	4	9
	无实践	*	*
设备	无	0	0
	标准	1	2
	专用	3	4
	特别	5	6

在 CEM 中对抵抗攻击的强度和 SOF 强度规定了 4 个取值范围，具体规定如表 30-5 所示。

表 30-5 安全强度分极表

范围	抵抗攻击的强度	SOF - 强度
< 10	不评估	不评估
10~17	低	基本
18~24	适度	中
> 25	高	高

2．安全功能的基准测评

安全功能的基准测评是商业领域的测评方法，例如，计算机领域中的 TPS 测试（测试每秒钟计算标准事务的数量）就是典型的基准测评。在信息安全领域，也广泛采用安全功能的基准测评，这种基准测评采用的是攻防技术基准测试。在一个应用模式结构中，可以固定防护技术，测试攻击技术；也可以固定攻击技术，测试防护技术；也就是说，基准测试是有针对性的。

3. 安全系统工程能力成熟模型评估

安全系统工程能力成熟模型（SSE-CMM, Security System Engineering-Capability Maturity Model）对信息系统工程项目和企业实施能力评估。

信息安全测评认证的实际工作重点包括：

- ◆ 信息安全产品、系统与服务的质量标准（CC 的 EAL 级别评估）
- ◆ 信息安全技术功能强度表（各种安全技术的功能强度的高、中、低描述表）

评价安全功能强度可以采用理论方法，得出对强度的不同认识。例如，对于访问控制，可以从理论上分析，强制访问控制的强度高于自主访问控制；又例如，对于加密算法，密钥长度小的算法强度低于密钥长度大的算法强度。评价功能强度还可以采用测试的方法。例如，在相同的攻击方法与相同的环境内，防护时间长的防护方法强度就高。所以，在实验室中可以采用基于时间的测试（TBS 测试）来判断安全功能强度。也就是说，给出安全功能强度表与给出攻防基准测试时间表，其意义是相同的。

4. 基于时间安全模型（TBS）的测试服务

主要是信息安全攻防技术的 TBS 测试基准表（固定攻击技术、测试攻击得手时间统计表以及固定防护技术、测试防护时间统计表）。

5. 其他信息化测评认证服务

除开展信息化安全测评认证之外，还应当积极开展如下的测评认证：

- ◆ 安全监管测评认证 安全监管测评认证的重要性是显然的，那么，如何对安全监管进行测评认证呢？监管测评认

证如果采用 CC 方法，其质量标准可以考虑采用 CC 准则，但是 CC 方法提供的第二部分（安全功能部分）将不适合安全监管测评认证。

- ◆ **安全应急（业务连续性）测评认证** 安全应急测评认证如果采用 CC 方法，其质量标准可以考虑采用 CC 准则，但是 CC 方法提供的第二部分（安全功能部分）将不适合安全应急测评认证。因为应急用到的技术在 CC 的第二部分中缺少相应的功能描述与规定，需要增加相应内容。当前可以采用或借鉴 FSA BCI 的标准进行测评认证或评估。
- ◆ **代理服务测评认证** 代理软件实现的网络远程服务。服务代理软件是指定功能的（不多做事情，也不少做事情），是可以测评认证的，从而便于管理、控制和计费。

笔者曾经建议修改 CC 标准，增加一些功能测评标准（信息技术安全测评认证准则）。例如，增加安全监管、安全应急和代理服务测评标准，构成新的安全测评认证标准，简称 New CC（其结构如图 30-4 所示），以适应代理技术时代的召唤和代理技术应用日益增长的要求。



图 30-4 New CC 准则的结构

30.2 测评认证的网络化服务

新世纪的测评认证方式是一种网络远程服务体系：建立会员制机构，建立标准的测评配置管理中心（SSA-CM），为会员单位提供服务。例如，信息技术厂商如果要为其开发的产品提供测评认证服务，需要向配置管理中心申报（这种申报往往是在 SSA-CM 的网站上注册）。经过研究与评估，如果企业的申请得到受理或批准，那么 SSA-CM 给厂商颁发入会上网的许可证及数字证书的相应身份，这样，厂商就可以从 SSA-CM 的网站上下载测评的整个生存期的各设计阶段（设计、开发、实现、测试、安装、配置、维护等）的标准、规范、文档、工具和核心软件实现等。厂商在进行规定范围之内的开发后，将甚至还是阶段性的产品递交配置管理中心，交测评机构进行测评，测评机构给出产品安全性和互操作性的级别评估。这种测评方法可以把测评服务大大前移，厂商从设计阶段到后期各阶段都可以得到测评认证服务。从总的测评时间来看，没有变化，但是，从产品最后得到认证的时间来看，大大前移了。由于在设计阶段测评工作就介入了，所以这种测评模式的测评质量和专业水平大大提高。同时，还可以评估信息技术厂商的能力成熟度模型（CMM）的水平。显然，这种新的测评模式会得到厂商的普遍欢迎。

测评认证的网络化服务是大势所趋，是信息技术企业实施服务低成本的主要措施之一。根据我国的特点，参考国外网络远程服务的管理办法，建立配置管理中心是一个好方案。我国的 SSA-CM 机构应当设置在测评认证中心，信息产业提供的网络远程服务必须通过 SSA-CM 管理监控，纳入政府管理产品类（GOTS）。测评认证机构提供的这些服务是有偿的，可以通过收

取会员费，有偿下载文件、标准、规范、工具和核心实现等（价格是国家规定的）来取得收益。在这方面取得成功的案例是美国国防部制定的 DII COE 计划的互操作性和一致性测评认证。

信息化配置管理体系是国家或者领域进行信息化标准控制的体系。信息技术与产品的市场准入制度在网络时代的控制，不是将市场准入的证书或者测评认证的证书挂在墙上，这仅仅是完成了所谓“法律概念”上的认证认可工作。

目前，国家的某些检测与测评机构热衷于权力意义上的批准程序，喜欢给企业和用户发证，但是它们忘了现代信息化测评认证工作是一种高水平的技术工作，对于人员素质和能力有很高的要求，这些能力要求突出表现在分级测评工作中，需要有“金刚钻”，需要进行实在的能力建设。整个体系提高测评业务服务能力十分重要，学习与培训任务艰巨。提高测评认证服务能力要通过开展更高水平的业务工作来推动。建立信息安全测评认证专业体系是提高测评认证服务能力的重要措施。必须建立芯片、硬件、操作系统、数据库管理系统、通信网络设备与系统、自动化控制设备和互操作性等专业领域的安全实验室，开发测试工具、配置工具、防护工具、攻击工具、监管工具、维护工具、诊断工具和监控工具等，同时还应具有信息安全教育与培训中心的职能。信息安全测评认证的配置管理体系将会对国家测评认证机构面向社会的前台服务提供基础支持，是信息安全测评认证的基础设施工程。应当提醒读者注意的是，测评认证机构实际上是指导用户建立信息化方案的机构，也是引导信息产业发展的机构，更是信息化的服务与管理机构。

另外，目前推出的测评认证模式也是传统的，对网络时代的测评认证、企业强烈要求的测评认证服务前移的呼声和标准化规定的测评认证的网络服务模式没有研究、缺乏认识或没有必要的

敏感性。笔者曾经多次给有关部门建议，建立国家、行业和领域的配置管理体系，实现和提高政府管理和指导信息化的能力（这也应当属于一种执政能力），但是很少有人重视该项建议。这使笔者想起一件事：几年前，为信息化建立互操作性平台时，笔者曾经对某人建议，信息化平台建设的核心内容是建立配置管理体系，此人并不以为然，但是过了两年，当他认识到这个问题时，却发现有许多人都明白了其中的道理，这时再去争取这个项目，就已经失去了机会。还有，笔者在几年前宣传代理技术和将代理技术研究从纯粹的人工智能研究中转移到信息化的主战场时，也遇到了很大的困难。有人认为，代理不就是黑客使用的小小软件吗？有什么好研究的？怎么还会有代理化？为此，笔者甚至讲出了“黑客的软件应用模式是先进生产力的代表”之类的话，来激励认识的改变。笔者的一些学生起初也不太理解，但是在笔者一再的督促下，他们现在认识到了研究代理技术的紧迫性，正在拼命研究代理技术各方面的进展。

最后，测评认证网络服务还要表现在测评认证产品的网上监控、监管和认证体系的建设上，确保每一个上网的产品都是经过测评认证的。在这方面，利用 CPK 组合密钥方法是一个好方案。

第 31 章

可信息化科学的形式化

31.1 计算是现代科学研究的第三支柱

这里谈到的可信息化科学主要指社会科学、军事科学、国家安全、自然科学、物理科学、天文学、生物科学、医学科学、地理科学、能源科学、环境科学、制造业科学等科学领域的信息化。在 2005 年 6 月，美国总统信息化咨询委员会（PITAC）提出了一个支持科学发展与创新的报告——《计算科学：确保美国的竞争力》（Computational Science: Ensuring America's Competitiveness）。这个报告中提到的一些观点值得我们认真参考。该报告提出计算科学继理论、实验之后，成为科学发展与创新的“第三支柱”，提出计算科学的主要动机是建立“计算基础设施”，强化更高能力、更大范围的复杂系统的计算能力，为美国的重要科学领域服务。在这个报告的社会科学部分，提出了美国经济监控、网络世界基础设施和社会科学、基于代理计算经济、政治社会科学档案等课题；在物理科学方面有：量子力学、高温超导模型、等离子和能源、粒子加速器、数据挖掘发现矮星、暗物质、暗能量和宇宙结构、超新星模型；在国家安全方面有：信号智能、人类环境实时复杂系统模型、传染病动态传播模型；在地理学方面有：风暴预警、加州地震模型与数据分析；在能源与制造业方面有：公路工程、可再生能源、石油资源地震找矿模型、涡轮发动机、造船业、航天航空高性能计算等；在生物科学与医学方面有：识别头脑疾病、蜜蜂通信与解码、蛋白质动力模型、蛋白质动力与功能、计算科学与医学护理等。

31.2 可信息化科学的形式化关系到人类 整个知识体系的世代传承伟业

在这里主要谈谈可信息化的问题，例如经济学、金融学、军事学和某些文化学等领域。我们研究社会行为实现信息化，长期以来是直接将社会科学的概念直接搬到计算机中，模拟人类在人类世界中的行为管理。虽然这种模拟体系在信息化初级阶段的建设中（信息化的历史才 20 多年）发挥了作用，但是在推进信息化的新发展中，这种模拟人类活动的信息化模式将不再会发挥重要作用，甚至还会阻碍信息化的发展。笔者想告诉读者的是，社会科学的信息化与形式化是本世纪最伟大的工作，它的难度、深度、全局性对整个人类发展的意义是十分重大的，甚至关系到人类整个知识体系的世代传承伟业。

不要以为当前开展的信息化工作已经到了一个高级阶段，信息化的发展已经不能仅仅通过“高速和宽带”发展模式来推动了，也不要以为可以仅仅通过需求来带动信息化的发展。最近，我们考察一些地方后发现，许多地区的信息化管理部门（包括相当高级别的信息化管理和推进部门）感到“不会搞信息化”，不知道信息化应当如何推动，推动什么。从理论与实际两个方面讲，当代的信息化仅仅处于初级阶段。

从前面的讨论中可以看出，信息化服务理论、信息化安全理论、信息化监管理论、信息化认证理论、网络行为对抗理论、信息化总体学与体系结构理论、信息化技术法规、信息化标准化理论、信息化评估理论、信息化测评认证理论等都需要研究人类信息化行为和网络世界虚拟主体的行为学理论，两个世界的行为学

理论研究是信息化社会科学最基础的理论研究。**对信息化社会科学与网络世界中的行为进行形式化的研究，是推动新一轮信息化发展的重要措施。**因为人类社会中的人类行为和网络世界中的主体行为已经构成了整个信息化社会科学的中心内容，所以这种观念不论从哲学上讲还是从事物发展的逻辑上讲，都是正确的。可以说，对人类世界中的人类行为（活动）、人类信息化行为（活动）和网络世界中的虚拟主体（代理）行为的研究，开创了以行为学为中心的形式化研究工作，它已经不是自动机、计算机、程序语言和系统之类概念的形式化定义的研究工作了。对人类世界中的人类行为（活动）和人类信息化行为（活动）两个方面的研究开辟了社会科学的形式化研究工作。在这方面，笔者愿意将在著作《软件行为学》中建立的行为描述的形式化方法推荐给读者作为借鉴。

前言中曾经谈到过，本书献给社会学家、经济学家、金融学家、管理学家、系统工程师、计算机科学家和数学家，把他们各自的方法学结合起来，把各自的关注点结合起来，去建立统一的认识体系平台。在这个认识体系平台上，社会学家、经济学家、金融学家、管理学家、系统工程师、计算机科学家和数学家各自实现新的方法与进行实践时，可以得到其他学科的支持，从而达到一个新的高度。

科学发展观要求经济发展与社会进步和发展相协调，要求社会发展与自然环境相协调。科学发展观首先要求实现科学决策和民主决策。这里不讨论民主决策问题，仅就科学决策进行讨论，它是一项非常复杂的工程。科学决策不仅需要具有不同视角的多方面的专家、人士协商，还需要决策支持的信息化系统，尤其需要规划决策的理论、模型、计算、实验、模拟、仿真、试点、测试、度量等多方面的支持。

笔者在南方一些大学讲课时，曾经考察过南方某著名大学的 CAD 或者虚拟现实的重点实验室和研究中心，在这个国家级别的研究中心中，建筑、工业、电子、生物、医学的相应模拟仿真系统与虚拟系统有相当大的成绩。但是，这个国家级的实验室和研究中心不提供社会、经济和生活领域的宏观规划信息化服务，例如城市交通规划、环境规划、资源规划、能源规划和市场模型等。笔者当时曾建议该中心的领导，积极开展这些国家急需的规划信息化服务建设，因为它们是国家与地方宏观决策非常需要的重要工作。另外，国家有许多规划研究与设计院，它们依然从事着土地、建筑、道路等基础建筑意义上的设施项目的规划与设计，基本上没有为国家、城市、领域提供信息化服务（即使有，也相当落后）。

例如，城市交通的仿真模拟系统的建设，在过去需要超级计算机系统，实行集中式的模拟仿真计算；但是现在，在适当规模的网络中，利用多代理技术实现几十万或几百万规模的城市交通的仿真计算，是完全可行的。图 31-1 显示了城市交通模拟仿真体系的结构。

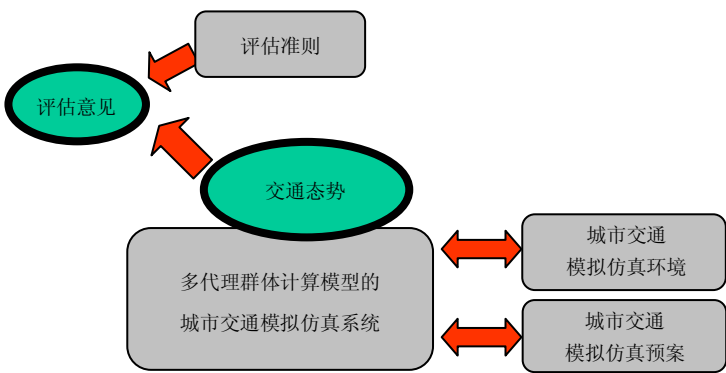


图 31-1 社会规划仿真模拟

其他方面的社会科学问题，例如社会学、经济学、金融学等方面的决策问题，必须通过建立一系列的规划决策信息化系统，才能逐步做到科学决策。在建立与发展规划决策模拟仿真系统时，社会科学的信息化与形式化也会得到提高，为产生社会科学信息化数学体系奠定基础。

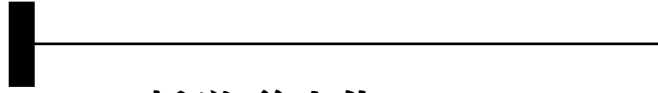
回过头来思考，中国人研究行为学具有最久远的历史，虽然那时不叫行为学，而叫做易经、道德经、礼仪、儒学和兵法等等。由于这些科学非形式化，不能反复试验和重演，所以必须仔细体会和领悟，才能理解它们。在这方面，社会科学与自然科学在人类接受的程度上差别极大。

对于自然科学的认识，通过教育已经或在相当程度上解决了知识的继承性和发展的问题，那些早期自然科学最前沿的成果中有些现在已经成为大学甚至是中学的教育内容，它们很容易被受教育者接受，因为受教育者对接受前辈的研究成果没有抵触情绪。

但是，对社会科学的认识与学习就没有这么简单了。在向学生教授社会科学的成果时，很容易受到抵触。虽然 2500 多年前提出的社会科学方法与结论依然被人们当做神圣的内容来传授，儒学、孙子兵法等都是如此；虽然也有人利用“三字经”等，强迫不懂事的孩子去背那些先人的研究成果与结论，但依然不能将人类社会科学的早期研究成果转变成学生可以理解的知识。所以，社会科学在认识论上有一个公理，这就是孔子的名言：**吾十有五而志于学，三十而立，四十而不惑，五十而知天命，六十而耳顺，七十而从心所欲，不逾规。**当人们体会到社会科学中的许多理论概念时，也已经到了老年。从出生到死亡，人对社会的认识就像锯齿波一样，随着出生、成长和死亡的周期而起伏。社会科学没有形成一个形式系统供人们反复试验，验证它的正确性（当然也是形式化正确的验证过程）。但是，社会科学教育与认识的可试

验和重演，在信息化和网络化的时代变得可能了，其出路就是不断地完善信息化社会科学的形式化。当然，这种形式化必须是计算机可读的。将现代科技与社会科学研究成果不断地传承下去，应当是整个人类的伟大工程，需要得到更多人的关注。

第 32 章



哲学形式化

32.1 网络世界的哲学体系

研究社会科学的形式化，其中很重要的是研究哲学的形式化。这个题目就是要给计算机提供能够读懂和理解的哲学体系，并为网络世界提供一个形式化的哲学体系。

在信息化和网络化的时代，人们可以提出社会科学的形式化研究课题。除了研究以行为学为中心的形式化理论方法外，还需要一个支持社会科学（而不是自然科学）的逻辑哲学体系。这种哲学体系与诞生于自然科学的那种追求真理的逻辑学（例如数理逻辑）是完全不同的，它是研究行为以及矛盾变化和演变的形式逻辑学，是计算机可读的。

笔者很早就对这种哲学形式化，尤其是对计算机可读的形式逻辑非常感兴趣。笔者建议进行两种逻辑学的研究：

其一，行为形式逻辑学。行为逻辑学可以是人类世界中人类行为的逻辑学，也可以是人类信息化行为的逻辑学，当然更可以是网络世界中虚拟主体行为的逻辑学。这个行为形式逻辑对应着中国哲学范畴体系的“人道”，研究“人与性”、“义理”、“公私”、“善恶”、“知行”、“理势”、“消长”等规范人类行为的论述。有志者可以研究之，为人类做出不朽的贡献。

其二，易理形式逻辑学。笔者在研究了类型程序设计方法学之后，于十几年前提出了易理形式逻辑学，也曾经试图将其作为一个正式的研究课题，并且开始了许多准备工作，研究一种内涵逻辑（而不是外延逻辑），研究事物变化的逻辑学。早在十几年前，笔者曾经与毕业于哲学专业的著名计算机科学家唐稚松院士就这个想法讨论过多次，我们都认为：只有对哲学实行形式化，才会有计算机的真正智能化。在 20 世纪 90 年代中期，笔者认识

到要想研究易理形式逻辑学，必须首先研究软件行为学。因此，积极进行软件行为的研究工作，经过 8 年的时间，终有了初步成果（汇集出版了《软件行为学》一书）。

32.2 中国哲学范畴最有希望发展为网络世界的哲学体系

中国哲学家喜欢把哲学发展描述成哲学范畴的发展。人文科学的哲学家们喜欢用“范畴”概念，而研究计算机科学或信息化科学理论的工作者也喜欢“范畴论”的知识。学习易经时，你自然会想到德国数学家莱布尼兹学习了易经，然后提出了二进制，所以也期望在其中找到一种简单的方法来描述矛盾转化的逻辑。如果你熟悉范畴论，那么就会知道，易经研究天道和人道的哲学范畴是一种建立在行为基础上的哲学与逻辑。中国哲学概念“阴阳”、“无极”、“太极”、“二仪”、“四象”、“八卦”、“阴与阳”、“一分为二”、“合二为一”、“中庸之道”、“对立统一”和“矛盾转化”等讲的是行为的互动态势与结果。行为中可以有对偶和非对偶的范畴。所以，为了研究这种逻辑形式体系，必须首先对行为形式化进行研究。在莱布尼兹所处的时代，不可能简单地从易经中找到易理形式逻辑学，因为研究行为学理论，如果不在信息化的当今时代，是不可能的。

支持中国传统社会科学发展的是中国的哲学范畴，而不是建立在西方自然科学基础上的逻辑学。中国的哲学是对人与自然进行统一认识，而西方哲学是将人与自然分离开来建立哲学范畴。西方自然科学的逻辑学基础是带动自然科学发展的强大动力和内

在因素。中国人的哲学中也有描述自然的，例如，老子的《道德经》反映了“天道”、“阴阳”、“无极”、“太极”、“二仪”、“四象”、“八卦”、“阴与阳”、“一分为二”、“合二为一”、“中庸之道”、“对立统一”和“相互转化”等许多哲学范畴。这种哲学思想更加适合建立和谐世界。哲学思想范畴从来都是被人们用文字形式相传至今的。如果人们只能不断去依照“三十而立，四十而不惑，五十而知天命，六十而耳顺，七十而从心所欲”的规律去研究这些哲学，那么社会科学的发展将永远是锯齿波发展形态，而不能被计算机所理解，也不可能成为网络世界中的哲学思想。

第 33 章

信息化科学使命

信息化是一种利用信息技术、产品与服务的社会活动的进程。信息化技术是运用 IT 技术与产品的再工程和再设计的技术。信息化科学是人类信息化行为科学与网络世界行为科学和计算机科学进行互动的科学。信息化科学涉及经典科学（电子学、磁学、光学、半导体学、电子计算机和光学计算机等），也涉及数学科学领域。例如，软件领域中主要涉及数学，这是传统计算机科学的研究领域；当然，信息化科学还涉及应用科学，这是当代信息化科学的研究重点。下面是笔者的一些看法，供读者参考，希望在信息化科学建立和发展的过程中得到不断成熟和完善。

一、应当建立一个什么样的信息化科学？

以研究人类世界的信息化行为和网络世界的行为为对象，建立和不断完善网络世界的行为学。从学科建设来说，主要包括以下几个方面：

- ◆ 网络行为应用模式理论（行为本体语义）。行为应用模式理论主要研究网络虚拟主体（代理）的行为，包括个体与群体的行为以及群体行为能力的扩展。另外，还要对行为结构、主体结构 and 客体结构进行研究，具体研究“谁做什么”的问题。其中，必须对网络行为保密、网络行为控制、网络行为监管、网络行为认证、网络行为对抗、代理执法、数字警察、信息战网络对抗、网络行为博弈等方面的理论进行研究。
- ◆ 网络行为形式逻辑学（行为规则语义）。研究网络行为规则结构理论，研究网络世界中虚拟主体（代理）的行为逻辑概念方面的问题，研究行为的规则结构，研究“怎么做”、“在什么地方做”、“在什么时间做”、“谁曾经做什么”、“谁

将要做什么”、“用什么方式做”和“能够做什么”、“不能做什么”、“必须做什么”等问题。使虚拟主体的行为更加理性和智能化，使网络虚拟主体具有目标性的行为分析、归纳和推理能力。

- ◆ 易理形式逻辑学（行为态势演化的语义或者行为造成的事务演化的语义）。研究行为结果的内涵逻辑和事物变化的逻辑。只有研究事务变化的逻辑，才可能真正进行行为的目的、企图和动机范畴的理论研究。否则，网络主体行为的目的、企图和动机是被人强加的。研究“为什么做”、“能够做什么”、“不能做什么”、“必须做什么”等问题，显然，这方面的研究比较困难。
- ◆ 行为模式的综合范畴语义。随着行为应用模式、逻辑和目的语义研究的深入，会产生行为模式、逻辑和目的综合范畴领域，这种行为综合范畴必然会产生国情、种族、文化、领域和社会的高层形式化的研究成果。
- ◆ 提出如何研究信息化科学的子科目，并为信息化科学本身的研究提出信息化理论、方法和模型。

信息化科学应当与计算机科学很好地结合，它们之间既有分别，又有交融。通过信息化科学带动计算机科学发展，计算机科学发展又为信息化科学发展提供新的基础。

二、信息化科学的目标是什么？

信息化科学是面向信息化实践与问题的，具有实际指导性，其首要目标是改变信息化混沌和没有理论指导的初级状态。信息化科学具有前瞻性和预见性，能够指明信息化的发展方向。要做到有预见性，必须对人类行为的可信息化问题有全面的认识，了

解与认识信息化发展阶段，对行为信息的数字化和可视化以及行为的代理化和智能化提出发展策略。

另外，信息化科学的发展对社会、政治、军事、经济、金融、哲学、文化、环境、资源和管理等科学研究具有推动力，会对人类行为学（包括社会行为学和组织行为学）的研究做出贡献。反过来，这些科学的发展也会对信息化科学提出更高的要求。

三、信息化科学当前研究的重点是什么？

当前，信息化科学以信息化目前面临的主要任务为研究重点：信息化如何科学发展？如何发挥信息化综合效益？如何加强信息化总体学与体系结构的指导作用？如何强化信息化公众服务、互操作性、安全、监管与认证？如何强化信息化技术法规、标准、评估和测评？所以，信息化科学要在群体计算理论、体系结构理论、互操作性理论、安全性理论、监管理论、认证理论、测评认证、评估理论、标准化理论和技术法规理论等方面有所突破和发展。

信息化科学研究需要进行总体结构的研究，提出和不断修改信息化科学的体系结构框架，确立和修改信息化科学的研究内容与科目。

四、信息化科学的研究方法是什么？

信息化科学研究不能仅仅在人类的头脑中进行，也不能仅仅将它理解为一种社会活动或人类的思考和推理活动，当然，也不能将它理解为一种实验室的活动。信息化科学研究必须实现本身的信息化，通过专门的信息化科学的信息化理论、方法和模型来研究信息化科学中的问题，使信息化科学活动、理论、方法与结论可以重演、传承和教育，使信息化科学教育与认识可试验和重

演。为达到此目的，应不断地完善信息化科学的形式化，而这种形式化必须是计算机可读的。

五、信息化科学的研究特点是什么？

由于信息化科学是新型学科，人们对它的认识还相当粗浅，所以需要不断深化认识信息化科学的特点。信息化科学研究与传统的自然科学（物理学、化学、生物学）和数学等理论研究有很大的不同。传统的自然科学有自己长期稳定的研究对象，对学科的深入探询与研究在相当大的程度上是指标性的；而新学科则是创造或等待技术条件的出现，然后通过不同方式去攀登研究的高峰。研究的方法包括理论论述、科学计算和科学实验。通过确立课题项目和建立项目组进行研究，学科的带头人是理论家或技术专家。理论成果的正确性是通过实验室反复实验得到证明的。信息化科学与数学也有较大的区别，数学是规避错误的，是通过演算与证明来论证事物是非的；而信息化科学天天与错误打交道，要承认错误，并且不断通过服务去修改错误（如果能够修改的话）。现代信息化学科的研究对象不是固定的，而是以信息化应用中提出的新问题作为研究对象的。新学科是在多学科融合中、在对理论与实践差距的认识中、在新应用模式的出现过程中提出的。研究的方法包括理论论述、模型建立、体系结构描述、方法学研究、软件实现和应用验证。信息化科学通过总体研究确立学科架构和课题项目。学科带头人包括理论家、技术专家、工程师、系统设计师和总体设计师。信息化学科的正确性和价值是通过应用来证明的。读者只要比较信息化学科与传统学科的不同点，就会对信息化学科有新的认识。

因此，我们要以新认识、新知识、新理论和新学说带动信息化发展，实现信息化发展的科学梦想！

附录 A

研究问题清单

在这里推荐 100 多个研究问题，供有志者进行研究。

1. 信息化科学梦的浮出

1) 了解计算机科学的发展历史。

- 2) 综合了解现代数学与计算机科学的关系。
- 3) 计算机需要可读的数学吗?
- 4) 深入理解信息化为什么需要形式化。

2. 计算理论

- 5) 学习可计算性理论。我们知道, 单带图灵机与多带图灵机在可计算方面是等价的, 那么, 无穷多带图灵机与单带图灵机在可计算方面还等价吗? 如果图灵机是一阶无穷计算机, 那么二阶无穷计算机或者更高阶无穷计算机应当是什么样的?
- 6) 温习算法理论和算法复杂性理论。
- 7) 了解算法信息论。
- 8) 了解算法的设计原则。

3. 电子计算机

- 9) 了解 LISP 计算机、组合逻辑计算机和 PROLOG 逻辑计算机当时的发展情况。
- 10) 为什么日本特别有兴趣研究机器人, 而美国的 IT 技术研究与应用更注重全局性和基础性, 例如研究代理技术和光学计算?
- 11) 描述一个几万、几十万、几百万、几千万或几亿个 CPU 的计算机封是如何构造的? 如何工作的? 到底应当如何构造如此规模的计算机? 如果你对计算理论有兴趣, 不妨研究关于高阶无穷计算机的计算问题。

4. 程序设计语言

- 12) 曾经有 300 多种程序设计语言, 为什么最后是 C 语言? 请熟悉 C 语言。

- 13) 了解 C 语言的编译程序和链接程序。
- 14) 熟悉 C++ 语言吗? 熟练使用 Class 技术和 Visual C++。
- 5. 计算机操作系统
 - 15) 论述代理网络操作系统将代理或多代理概念作为网络系统的核心概念与传统的操作系统以进程为核心概念的区别是什么? 它如何支持几万、几十万、几百万、几千万或几亿个代理软件运行?
 - 16) 不同计算机应用需要不同特点的操作系统, 了解几种不同技术体制特点的操作系统。
 - 17) 从理论内部体系结构讨论代理网络操作系统的原理与实现技术体制。
- 6. 计算机应用
 - 18) 了解计算机应用中管理信息系统(MIS)的发展过程。
 - 19) 了解计算机应用中管理系统(MS)的发展过程。
 - 20) 对未来计算机应用的发展如何展望? 为什么会有 Agent MIS?
- 7. 网络系统与网络世界
 - 21) 了解电信、计算机和电视网络的发展史。
 - 22) 一网多用(综合业务)和多网融合在发展指导思想上有什么区别?
 - 23) 信息基础设施网络在考虑了网络安全之后, 网络的运营、系统和技术体制应当做什么根本性变化? 一个可信的网络基础设施体系结构的基本要求是什么?
- 8. 软件工程与面向对象程序设计
 - 24) 同样都是 OOP 方法, 支持软件重用的成本差别很大, 设计并实现常用类型的类型库, 设计系统服务的类型库和选择领域的业务类型库。

- 25) 为类型表达式定义一个说明语言, 并为该语言写一个编译程序。
 - 26) 为类型描述定义一个说明语言, 并为该语言编写一个编译程序。
 - 27) 设计一个 TOP 的软件自动生成系统。
 - 28) 为程序设计语言定义一套符号处理、项重写系统, 并为该系统提供逻辑类型(例如基本逻辑类型、模态逻辑类型、时态逻辑类型、模糊逻辑类型、多值逻辑类型等), 体会软件智能化的真正含义。
 - 29) 为各种行为应用模式的形式逻辑体系定义类型, 把逻辑能力提高到各种领域的业务应用中。
 - 30) 一个考虑安全的软件工程应当是什么样的? 再造安全软件工程理论体系。
9. 并发程序设计与面向代理(主体)的程序设计
- 31) 熟练掌握传统的并发程序设计的方法。
 - 32) 什么是面向代理(主体)的程序设计?
 - 33) 尝试用类型方法定义并发程序设计的所有类型库, 并在此基础上定义各种领域概念的规范代理、代理集合、代理网络、多代理组织、代理网格平台和代理网格的类型。
 - 34) 为什么说如果仅有面向对象的技术, 而没有面向代理(主体)的软件开发技术, 我国软件产业则少了半壁江山?
10. 可视化与多媒体
- 35) 可视化技术是计算的理解技术吗? 为什么所有的软件都可可视化了?
 - 36) 为什么说可视化技术的提出是计算机发展的第二个

里程碑，并带动了计算机技术快速发展？

- 37) 在计算机科学家期待的虚拟现实的研究进展不大的时候，可视化技术为电影、电视、动画和游戏在网络上的传播带来了空前的发展，可视化的 IT 技术的内容建设支持着互联网的广泛应用。请思考这种现象。

11. 光学计算与光学计算机

- 38) 找一本光学计算机的原理书认真学习。
- 39) 设计一个光学计算的多维模型，并为该模型选择逻辑体系。
- 40) 构想或设计一个光学计算机的程序设计语言。
- 41) 构想或设计一个光学计算机的操作系统。

12. 计算的理解科学（形式语义与形式说明）

- 42) 你学习过形式语义学吗？如果没有，找一本书学习。

13. 系统集成

- 43) 系统学习 Microsoft 公司的 OLE/COM/DCOM 部件对象技术体系。
- 44) 系统学习美国国防部的公共操作环境（COE）段开发技术。

14. 统一建模语言（UML）与可扩展标记语言（XML）

- 45) 了解 OMG 组织的工作和已经取得的成绩，尤其要关注 CORBA。
- 46) 系统学习 UML 2.0 和 XML 的家族语言。
- 47) 尝试使用 UML 的软件生成工具，并做出自己的评价。
- 48) 了解语义 Web、DAML 和 Agent UML 的进展情况。

15. 代理化

- 49) 论述代理技术的本质是代理性。
- 50) 论述代理的自治特性是必然的要求。

- 51) 论述代理的人工智能特性要符合时代的可行性要求。
- 52) 论述代理是面向有组织群体的协同技术。
- 53) 论述建立可生长和可移动的代理群体体系结构是多代理技术追求的建立不确定计算的目标。
- 54) 了解 FIPA, OMG 和 RFC 等组织的代理技术标准与规范。
- 55) 构思、设计并实现规范代理, 尝试行为踪迹的再执行, 体会代理行为行为树、行为信息基的意义。
- 56) 构思、设计并实现代理集合。
- 57) 构思、设计并实现多代理系统。
- 58) 构思、设计并实现代理网格平台。
- 59) 构思、设计并实现代理网格的框架。
- 60) 如何建立 Agent Factory 的产业模型体系?

16. 互操作性平台理论

- 61) 什么是互操作性? 研究各种引用实现技术及其特点。
- 62) 了解已知的互操作性引用实现技术, 看看它们之间有什么共同的规律性?
- 63) 设计一个面向多系统和多代理的平台。
- 64) 设计一个面向可视化与代理化服务的平台。
- 65) 研究互操作性引发的安全问题。
- 66) 研究远程服务的安全性问题。
- 67) 研究互操作性的测评认证问题。
- 68) 学习 DII COE 的互操作性测评认证方法。
- 69) 如何采用代理技术实现互操作性?

17. 网络世界行为学

- 70) 分析如何对网络世界的行为进行语义描述。
- 71) 为什么要提出研究软件或网络行为的信息综合要

求、行为模式的应用要求、行为特性的能力要求和行为平台的系统要求？如何定义行为语义模式的本体语义？

72) 确立网络世界行为的状态转换模式。

73) 确立网络世界的行为特性（例如行为可信性、有效性、完整性、保密性和连续性等）。如何描述与定义这些行为特性？如何定义行为模式的规则（逻辑）语义？如何定义行为模式的安全语义？如何定义行为模式的态势语义？

74) 研究与讨论网络世界中虚拟主体行为的逻辑学框架与模型。

75) 如何定义行为模式的平台语义（类型指称语义）？

76) 研究网络行为逻辑学和网络行为保密学。

77) 研究网络行为监管与认证学。

78) 研究网络行为对抗学。

18. 信息化服务性理论

79) 结合一个领域（例如银行），给出业务与技术服务行为的逻辑形式化模型。

80) 为什么要搞软件支持活动配置管理中心（SSA-CM）？

81) 论述信息化的直接服务、网络访问服务、网络远程访问服务和网络代理服务的特性。

19. 信息化安全性理论

82) 结合一个领域（例如银行），给出业务与技术行为控制的逻辑形式化模型。

83) 为什么要划分安全保障、安全监管、安全应急和安全威慑 4 个安全工作范畴，而不是把所有的信息化安全统一在一个超级安全保障体系中？

- 84) 如何实现计算机可信计算基 (TCB) 的多代理扩张?
为什么采用活性标签技术和 Agent-MAC? 为什么有了代理的个体完整性后还需要群体的完整性?
- 85) 如何实现计算机的 TCB 与 TCP 的一体化?
- 86) 对于 TCP 而言, 如何实现多厂商的 TPM 互操作性?
- 87) 对于 TCP 而言, 如何实现 TPM 与外延设备的代理化扩展?

20. 信息化监管理论

- 88) 选择一个领域, 定义信息资产价值计算公式和风险价值计算公式。
- 89) 设计并实现一个方法监管的多代理体系。
- 90) 设计并实现一个结构方法监管的多代理体系。
- 91) 设计并实现一个面向业务行为与内容的监管多代理体系。
- 92) 设计并实现一个面向技术行为与内容的监管多代理体系。
- 93) 设计并实现一个结构行为监管多代理体系。
- 94) 设计并实现一个多结构行为监管多代理体系平台。
- 95) 结合一个领域 (例如银行), 给出业务行为与内容监管的逻辑形式化模型。

21. 信息化认证理论

- 96) 根据组合密钥体系思想, 能否提出一种新组合密钥算法?
- 97) 设计并实现一个活性密钥体系。
- 98) 设计并实现基于多代理技术体制的认证体系, 实现认证代理化。
- 99) 活性认证为 CPK 密钥管理算法提供什么安全可信机

制可以保证密钥的安全性？为什么有了认证代理的个体完整性后还需要群体的完整性？

- 100) 结合一个领域（例如银行），给出业务与技术主体、客体、行为与内容认证鉴别的逻辑形式化模型。

22. 网络行为对抗理论

- 101) 实现一个可移动、可生长的多代理体系的数字化师（团）。
- 102) 研究网络行为对抗的科研、设计和生产体系。
- 103) 建立一个面向网络行为对抗的 Agent Factory 的产业模型体系。
- 104) 研究网络行为博弈理论。
- 105) 研究代理生存与消除对手的方法。
- 106) 研究模式发现与模式隐藏以及行为发现与行为隐藏方法。
- 107) 研究行为控制与反控制方法。
- 108) 研究行为特性对抗方法。
- 109) 研究网络对抗的快速有线插播攻击的原理与方法。
- 110) 研究网络对抗的快速卫星通信攻击的原理与方法。
- 111) 研究网络对抗的快速无线通信攻击的原理与方法。
- 112) 研究定位与反定位以及追踪与反追踪的原理与方法。
- 113) 研究行为对抗组织输送和配置的理论与方法。
- 114) 研究行为对抗能力评估（红/蓝测试）的原理与方法。
- 115) 研究代理执法学。
- 116) 研究数字警察学。
- 117) 研究数字隐蔽对抗和网络恐怖对抗。
- 118) 研究信息战网络对抗学。

23. 群体计算理论

- 119) 试证明群体计算无限资源假说是正确的。
- 120) 试证明可生长、可移动群体计算无限资源假说是正确的。
- 121) 试证明可生长、可移动群体不确定计算命题是正确的。
- 122) 试证明群体计算模型的生存能力比传统计算模型强。
- 123) 试证明群体计算模型的预警能力比传统计算模型强。
- 124) 试证明群体计算模型的大范围分析与处理能力比传统计算模型强。
- 125) 给出多主体多行为的相关性分析研究结论。

24. 人类信息化活动行为学

- 126) 为什么要提出研究人类信息化行为的信息综合要求、行为模式的应用要求、行为特性的能力要求和行为平台的系统要求？
- 127) 构想和设计人类信息化活动的主体结构、行为结构、客体结构和规则结构。
- 128) 讨论、设想、构思和尝试研究人类信息化行为逻辑学。

25. 信息化总体学与体系结构

- 129) 为什么说人类信息化行为与网络世界行为是研究信息化体系结构的基础？
- 130) 研究运营范畴的主体结构、行为结构、客体结构和规则结构理论。
- 131) 研究系统范畴的主体结构、行为结构、客体结构和规则结构理论。

26. 可信网络世界体系结构框架（TCAF）

- 132) 为什么要搞可信网络世界体系结构框架（TCAF）？

- 133) 设计一个可信计算平台 (TCP) 体系结构。
- 134) 设计一个可信网络平台 (TNP) 体系结构。
- 135) 设计一个可信管理平台 (TMP) 体系结构。
- 136) 设计一个可信认证平台 (TCAP) 体系结构。
- 137) 设计一个可信应用风险监管平台 (TSP) 体系结构。
- 138) 设计一个可信应用平台 (TAP) 体系结构。
- 139) 设计一个可信信息交换平台 (TXP) 体系结构。
- 140) 设计一个可信交易平台 (TECP) 体系结构。
- 141) 设计一个可信公众服务平台 (TPSP) 体系结构。
- 142) 设计一个可信数据库访问管理平台 (TSHADEP) 体系结构。

27. 信息化技术法规

- 143) 研究技术法规为什么要研究人类信息化行为学与网络世界行为学的基础理论?
- 144) 尝试以人类信息化行为学与网络世界行为学为指导, 制定一个网络代理法和代理执法的技术法规体系模型。
- 145) 分析已有的各种信息化技术法规中存在的问题, 并提出自己的修改意见或方案。

28. 信息化标准化理论

- 146) 什么是用户标准化体系?
- 147) 什么是信息化的体系结构标准?
- 148) 什么是信息应用平台标准体系 (互操作性标准)?
- 149) 什么是信息安全标准体系?
- 150) 什么是网络技术服务标准体系?

29. 信息化评估理论

- 151) 为什么要紧紧抓住信息资产价值评估和风险价值评估?
- 152) 为什么要对信息资产价值评估进行有效性的再评估?
- 153) 评价当前流行的评估方法存在的问题, 并提出修改意见或方案。

30. 信息化测评认证理论

- 154) 了解 TCSEC 测评标准中存在的问题。
- 155) 了解 CC 测评认证标准和公共测评方法。
- 156) 了解 SSE-CMM 和 ISO 17799 评估方法中存在的问题, 并提出修改意见或方案。
- 157) 为什么要搞测评认证服务配置管理体系? 提出一种测评认证前移和进入网络的服务体系。

31. 可信息化科学形式化

- 158) 怎样理解当代信息化从软件的智能化角度看仅仅处于初级阶段?
- 159) 为什么要做社会科学的信息化与形式化工作?
- 160) 利用多代理群体计算方式构思和设计城市交通模拟仿真系统。
- 161) 利用多代理群体计算方式构思和设计资源利用模拟仿真系统。
- 162) 利用多代理群体计算方式构思和设计环境保护模拟仿真系统。
- 163) 利用多代理群体计算方式构思和设计市场竞争模拟仿真系统。

32. 哲学形式化

- 164) 效仿数学家莱布尼兹学习易经, 看看有什么体会。
- 165) 研究人类行为的逻辑学。

166) 研究人类信息化行为的逻辑学。

167) 研究网络世界的虚拟主体的行为形式逻辑学。

168) 研究易理形式逻辑学。

33. 信息化科学

169) 什么是信息化科学? 为什么要建立信息化科学?

170) 论述信息化科学与计算机科学之间的关系。

附录 B



资 源

1. Foundation for Intelligent Physical Agents (FIPA),
<http://www.fipa.org/spec>
2. Distributed Management Task Force (DMTF),
<http://www.dmtf.org>

3. IETF, <http://www.IETF.org>
4. W3C Web Services, <http://www.W3C.org>
5. OMG Agent Working Group, <http://www.omg.org>
6. DOD, U.S. Department of Defence
7. DOE, U.S. Department of Energy
8. ISO, <http://www.ISO.org>
9. ITU, <http://www.ITU.org>
10. COPC, <http://www.COPC.org>
11. EPRI, <http://www.epri.com>

参考文献

- [1] 屈延文. 软件行为学. 北京: 电子工业出版社, 2004.
- [2] 屈延文, 南相浩, 韩纬玺, 王永红, 林鹏, 王贵驷等. 银行行为监管——银行监管信息化. 北京: 电子工业出版社, 2004.
- [3] 屈延文, 南相浩, 韩纬玺, 王永红, 林鹏, 王贵驷等. 银行行为控制——银行信息化与安全. 北京: 电子工业出版社, 2004.
- [4] 屈延文, 南相浩. CPK 活性认证体系与国家组合公钥基础设施 (NCI). 2005.
- [5] 屈延文, 李鸿培, 李建清. 活性安全标签与基于多代理技术的强制行为控制模型 (Agent MAC). 2005.
- [6] 屈延文, 李鸿培, 李建清. 网络行为监管的多代理模型. 2005.
- [7] 南相浩, 陈钟. 网络安全技术概论. 北京: 国防工业出版社, 2003.
- [8] 孙玉. 电信网络安全框架. 2005.
- [9] 孙玉. 电信网络总体概念. 2005.
- [10] 屈延文. 形式语义学基础与形式说明. 北京: 科学出版社, 1989 (第一次印刷), 1999 (第二次印刷).
- [11] 屈延文等. 实用类型程序设计. 北京: 科学出版社, 1992.
- [12] QNS 工作室, 北京天融信科技有限公司, 中国信息产业商会信息安全产业分会. 信息化体系结构方法. 2005.
- [13] QNS 工作室, 北京天融信科技有限公司, 中国信息产业商会信息安全产业分会. 可信网络世界体系结构框架 (Trusted Cyber Architecture Framework). 2005.
- [14] 毛新军. 面向主体的程序设计, 长沙: 国防科技大学出版社, 2005.

- [15] 中国信息产业商会信息安全产业分会. 我国信息安全产业调研与建议. 2002.
- [16] 中国信息产业商会信息安全产业分会. 中国信息安全产业发展白皮书. 2005.
- [17] 钟旭, 洪泽勤等. 椭圆曲线组合公钥在 IBE 加密体制中的应用研究. 2003.
- [18] Institute of Electrical and Electronics Engineers. *IEEE Std 1471-2000*. 2000.
- [19] Putman J. *Architecting with RM-ODP*. 2000.
- [20] Federal Chief Information Officers' Council. *Federal Enterprise Architecture Framework, Version 1.1*. 1999.
- [21] Bass L, Clements P, Kazman R. *Software Architecture in Practice*. 1998.
- [22] Bosch J. *Design and Use of Software Architectures*. 2000.
- [23] Buschmann F, Meunier R, Rohnert H, Sommerlad P, Stal M. *Pattern-Oriented Software Architecture. Volume 1: A System of Patterns*. 1996.
- [24] 美国总统信息化咨询委员会 (PITAC) 2005 年 4 月提出的报告: *Cyber Security : A Crisis of Prioritization*
- [25] Horst Herrlich, George E. Strecker. *Category Theory*. 1979.
- [26] H.K.Curry, R. Feys. *Combinatory Logic*. 1958.
- [27] Martin_Löf. *An Intuitionistic Theory of Type*. 1975.
- [28] Martin_Löf. *Constructive Mathematics and Computer Programming*. 1979.
- [29] Martin_Löf. *Intuitionistic Type Theory*. 1984.
- [30] Martin_Löf. *Mathematics of Infinity*. 1990.
- [31] O. J. Dahl, E. W. Dijkstra, C. A. R. Hoore. *Structured Programming*. Academic Press Inc, 1972.
- [32] E. W. Dijkstra. *A Discipline of programming*. New York, 1981

- [33] D. Bjarner, B. Jones. *The Vienna Development Method: The Meta-Language*
- [34] C. A. R. Hoare. *Communication Sequential Processes*. 1978.
- [35] Austin. *How to Do Things with Words*. Clarendon Press, 1962.
- [36] Searle J. R. *Speech Acts*. Cambridge University Press, 1969.
- [37] Cohen P. R, Levesque H. J. *Intention is choice with commitment*. Artificial Intelligence, 1990, 42 (2~3) .
- [38] Cohen P. R, Levesque H. J. *Communicative actions for artificial agents*. Proceedings of the First International Conference on Multi-agent Systems(ICMAS'95). San Francisco, CA, 1995.
- [39] The Hewlett-Packard (HP), HP-UX CMW, 1995.
- [40] The Hewlett-Packard (HP), Virtual Vault, 1997.
- [41] The Hewlett-Packard (HP), Digital MLS+, 1997.
- [42] The Hewlett-Packard (HP), SEVMS, 1997.
- [43] The Hewlett-Packard (HP), OPENVMS, 1997.
- [44] Microsoft Windows NT V4.0
- [45] CISCO, CIC, 2000.
- [46] Department of Defense. *Joint Technical Architecture Version 3.1*. 2000.
- [47] DII COE: Common Operating Environment (公共操作环境), Version 4.7. 2002.
- [48] DOD 可信计算机系统评估准则 TCSEC
- [49] CC COMMON CRITERIA, ISO/IEC 15408 信息技术安全评估准则